

## Trabajo Fin de Grado

Diseño e implementación de infraestructura cloud  
para el soporte del proyecto educativo en IoT  
"Pájaros en la nube"

*Cloud infrastructure design and implementation to support the  
IoT based educational project called "Birds on cloud"*

Autor/es

Víctor Delgado Alejandro

Director/es

Enrique Torres Moreno

Escuela de Ingeniería y Arquitectura / Campus Río Ebro

2019



## Resumen

El proyecto “Pájaros en la nube”, impulsado por Ibercivis y eTopia, tiene como objetivo involucrar a alumnos de primaria y ESO, y profesores de Biología y Tecnología, en un experimento de ciencia ciudadana que les permita estudiar la fauna insectívora y su hábitat (temperatura, humedad, presión). Este estudio busca concienciar de la desaparición de estos animales y apoyándose en paneles de control de datos, un primer contacto con el mundo del tecnológico y el Internet de las Cosas.

Participan además, la Escuela Superior de Diseño de Aragón, encargada de diseñar y fabricar la caseta y Enrique Torres, encargado de la electrónica y sensorización IoT.

Para llevar los datos recogidos por la caseta a internet, el proyecto se apoya en la plataforma The Things Network (TTN) en Zaragoza. Se necesita un conjunto de servicios que permita a profesores y estudiantes hacer uso de la información generada.

Tras el estudio de alternativas, se implementará un diseño que permita a los usuarios explotar la información desde cualquier parte, asegurando la integridad de la información y la seguridad del entorno.

TTN se integra con diferentes servicios entre los que se encuentra Amazon Web Services. Para elegir una implementación adecuada para el estado actual del proyecto se realizan 4 diseños a estudiar. Dos de ellos tienen como finalidad el uso de servicios especializados en IoT en la Cloud de Amazon como AWS IoT, otro diseño estudia el uso de una implementación basada en servicios estándar de AWS y el último que propone un diseño más tradicional basado en una máquina virtual administrada por Ibercivis.

Para evaluarlos se analiza la complejidad, el coste de tiempo y dinero, el potencial y los plazos del curso escolar.

En base a esos compromisos se concluye que cualquiera de los diseños implementados sobre servicios Cloud supone un coste de tiempo y dinero que escapa a la magnitud del proyecto. A pesar de perder el potencial que tiene la Cloud, para el estado actual del proyecto la mejor solución es implementar los servicios que cumplirán los requisitos de los usuarios sobre una máquina de Ibercivis.

Para los requisitos de los alumnos y profesores, visualización fácil e intuitiva, se implementa una integración directa entre TTN y la aplicación en la nube Cayenne.

Para los requisitos establecidos por los gestores del proyecto se implementa una stack de servicios que permite almacenar la información de forma persistente, controlar el flujo de datos de los mensajes y monitorizar mediante paneles de alarmas con notificaciones el estado de los dispositivos. Esta implementación se completa con un servicio que permite el aislamiento de los anteriores tras una capa de seguridad que permite utilizar tráfico HTTPS manteniendo la integridad de la información en un entorno público.

Tanto la aplicación de los colegios como la de gestión se encuentran en producción.



# Índice

1.	Introducción y objetivos .....	2
1.1.	Motivación .....	2
1.2.	Objetivos .....	3
1.3.	Descripción del documento .....	4
2.	Definición y análisis de Requisitos .....	6
2.1.	Arquitectura .....	6
2.2.	Usuarios .....	7
2.3.	Requisitos .....	8
3.	Análisis de elementos del diseño .....	10
4.	Estudio de los diseños .....	12
4.1.	Integración TTN – AWS IoT .....	12
4.1.1.	Diseño .....	12
4.1.2.	Pruebas .....	13
4.1.3.	Conclusiones .....	14
4.2.	Integración TTN – AWS IoT (Versión 2) .....	14
4.2.1.	Diseño .....	14
4.2.2.	Pruebas .....	16
4.2.3.	Conclusiones .....	17
4.3.	TTN – AWS Free Tier .....	17
4.3.1.	Diseño .....	17
4.3.2.	Pruebas .....	19
4.3.3.	Conclusiones .....	19
4.4.	Conclusiones AWS .....	20
4.5.	Diseño final .....	20
4.5.1.	TTN – Cayenne (Requisitos Usuarios) .....	21
4.5.1.1.	Formato de la trama .....	21
4.5.1.2.	Pruebas .....	24
4.5.1.3.	Conclusiones .....	24
4.5.2.	Stack IoT (Requisitos de los gestores) .....	25
4.5.2.1.	Diseño .....	25
4.5.2.2.	Pruebas .....	27
4.5.2.3.	Conclusiones .....	27
5.	Implementación .....	28
5.1.	TTN - Cayenne .....	28
5.2.	Stack IoT .....	29
6.	Validación .....	36
7.	Conclusiones .....	40
8.	Bibliografía .....	44
ANEXO	.....	46



# 1. Introducción y objetivos

## 1.1. Motivación

Pájaros en la nube es un proyecto impulsado por Ibercivis y Etopia que tiene como fin involucrar a colegios de primaria y ESO en un experimento de ciencia ciudadana.

El objetivo es permitir el estudio de los animales insectívoros y sus hábitats, para concienciar de la desaparición de estos en las ciudades con la ayuda de una caseta de pájaros sensorizada. Se busca involucrar tanto a profesores de Ciencias y Tecnología como a alumnos de un amplio rango de edades en un proyecto que abarca el estudio de los animales y de su hábitat, la construcción de la caseta y búsqueda de una localización adecuada y la observación de la fauna.

Además, se podrán estudiar datos del hábitat como la temperatura, humedad, presión o presencia. Aplicar inteligencia sobre esos datos para estudiar máximos, mínimos y medias será otra de las posibilidades así como la monitorización de los animales y la creación de alarmas para notificar, por ejemplo, con un mail cuando el pájaro se encuentre en la caseta.

En el proyecto participan además de los colegios, la Escuela Superior de Diseño de Aragón, encargada de la fabricación de la caseta y de adaptarla a cada colegio para la especie a estudiar y Enrique Torres, encargado de la sensorización.

La caseta mostrada en la Figura 1, es alimentada con baterías por lo que se busca reducir el consumo de los sensores para prolongar la vida de estas.



**Figura 1.** Caseta de pájaros sensorizada

Esto se consigue mediante la ejecución de una serie de pasos de manera cíclica:

- Despertar cada 10 minutos
- Tomar datos de los sensores
- Formar una trama con los datos y el estado del dispositivo

- Encolar y transmitir esta trama
- Esperar una ventana de tiempo a la recepción de algún mensaje
- Dormir

Las casetas transmiten los datos a través del Internet de las cosas o IoT[0] apoyándose en la plataforma The Things Network (TTN)[1].

TTN es una organización que tiene la intención de crear una red distribuida y descentralizada de IoT de código abierto. Esta organización basa sus comunicaciones actualmente sobre LoRaWAN (LongRange WAN) [2], esta tecnología utiliza frecuencias de radio para comunicar datos permitiendo ampliar el rango de alcance añadiendo puertas de enlace o gateways que hagan de punto de entrada a la red para los dispositivos. Esto les permite tener un bajo consumo, poder reducir costes al utilizar un ancho de banda bajo y debido al largo alcance, reducir el número puertas de enlace.

TTN solo se encarga de hacer llegar los datos de nuestros dispositivos a internet, será tarea del proyecto, estudiar diseños que permitan que los datos lleguen a los usuarios finales.

## 1.2. Objetivos

Como hemos visto, el objetivo de este proyecto es conseguir que niños de diversas edades puedan estudiar la fauna insectívora de Zaragoza a la vez que tienen un primer contacto con el mundo del IoT a través de paneles de control en la nube e impulsar el uso de TTN Zaragoza con la colocación de 3 antenas para crear la infraestructura.

Para conseguir esto es necesario también involucrar a parte del profesorado que no tiene por qué ser técnico, lo que podría sentar bases tecnológicas sobre estos perfiles dedicados a otras áreas.

La visualización de los datos de las casetas en un entorno sencillo para alumnos y profesores con bajos conocimientos tecnológicos ha sido uno de los objetivos que busca cumplir el proyecto.

La gestión de los datos de todos los dispositivos, así como el control del flujo de datos y el almacenamiento han sido objeto de estudio durante la evolución del proyecto.

El proyecto debe estar en funcionamiento en mayo debido a que los colegios tienen que desarrollar las diferentes actividades ligadas al proyecto durante este último trimestre.

Para cumplir estos objetivos se han estudiado diversos diseños se han ido descartando tras la implementación de una demo y de la evaluación de costes, complejidad, potencial y viabilidad.



El proyecto comienza en la asignatura de Laboratorio de Sistemas Empotrados con la toma de unos prerrequisitos y la decisión de usar AWS IoT con el fin de aprender la herramienta.

A fecha de hoy, el proyecto ha llegado a 10 colegios. Del total, 6 han comenzado ya con la parte de la asignatura dentro de la que se incluye este proyecto, los restantes o aún no han llegado al temario o no tienen cobertura TTN. De los 6 dispositivos, solamente 3 están conectados y enviando datos.

La gestión de los datos, almacenamiento, control de flujo y visualización de los datos globales, ha quedado integrado en una máquina de Ibercivis con IP Pública a la que se le han aplicado las medidas de seguridad necesarias.

La visualización de los datos de cada colegio ha sido integrada con Cayenne My Devices.

### 1.3. Descripción del documento

El documento está estructurado de la siguiente forma.

- El capítulo 2, explica la arquitectura del sistema que permite a los dispositivos llegar a internet, explica la existencia de dos grupos de usuarios con requisitos diferentes y establece los requisitos funcionales y no funcionales de la infraestructura a diseñar para hacer llegar esos datos de las casetas a los diferentes usuarios.
- El capítulo 3, expone los tipos de herramientas y las características que van a ser necesarias para llegar a cumplir los requisitos establecidos en el capítulo 2.
- El capítulo 4, presenta el estudio y la evolución de diferentes diseños que concretan y justifican el uso de herramientas o aplicaciones para alcanzar el objetivo del proyecto evaluando los requisitos establecidos en el capítulo 2 y exponiendo unas conclusiones.
- El capítulo 5, detalla la implementación de los diseños finales.
- El capítulo 6, verifica y valora el cumplimiento de los requisitos iniciales sobre la implementación final.
- El capítulo 7, expone las conclusiones generales del proyecto así como un diagrama de Gantt.
- Por último, el anexo, donde se adjuntará un manual de uso realizado para los colegios.



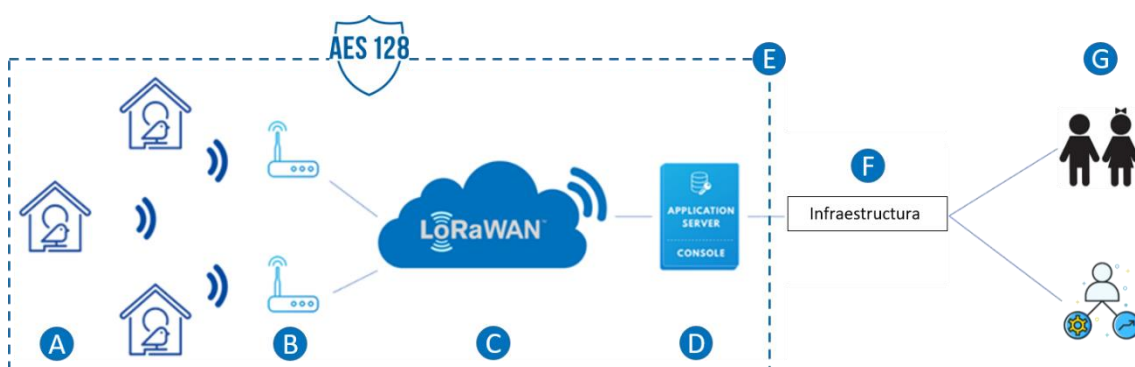
## 2. Definición y análisis de Requisitos

### 2.1. Arquitectura

Para estudiar el hábitat y la presencia de animales las casetas o dispositivos están dotados de tres sensores, dos que miden la presión, humedad y temperatura para recopilar datos tanto de fuera como de dentro de la caseta y un láser de alta precisión que se colocara dentro para detectar cuando el animal está en la caseta.

La recolección de los datos se realiza para apoyar el estudio de los alumnos y profesores del hábitat del animal a través de gráficas y paneles de control accesibles desde cualquier punto.

Para poder disponer de estos datos en internet es necesario la existencia de una plataforma con una arquitectura como la de TTN representada en la Figura 2.



**Figura 2.** Arquitectura TTN

Para que los mensajes de las casetas o dispositivos representados en la sección A de la Figura 2 lleguen a TTN, se insertan los datos en una trama con un formato determinado o payload y se envían mediante la tecnología de radio LoRa [3], la cual permite grandes alcances. El dispositivo transmite los datos y los gateways, representados en la sección B, que se encuentren dentro de su zona de cobertura los recogerán.

Los gateways se encargan de pasar la trama con los datos por internet a TTN, una red LoRaWAN, representada en la sección C, que es básicamente una arquitectura para redes de baja potencia y amplia área.

Para la gestión de los dispositivos y validación de los mensajes se utiliza la aplicación TTN, representada en la sección D. Se asocia un ID y un EUI únicos a los dispositivos, se recogen todos los mensajes asociados a dispositivos que estén registrados en ella y se filtran, eliminando repetidos y comprobando la integridad de los mensajes.

Para tener un control de accesos, en la aplicación TTN se definen usuarios colaboradores a los que se les asignan unos permisos sobre la aplicación TTN y los

dispositivos registrados en ella permitiendo granular los accesos y segmentar las acciones de cada usuario según convenga.

Para poder ver los dispositivos registrados así como los mensajes que van llegando, los usuarios pueden acceder a la administración de estas aplicaciones a través de una consola web representada también en la sección D. Además esta aplicación TTN ofrece la posibilidad de integrarse con diversos servicios.

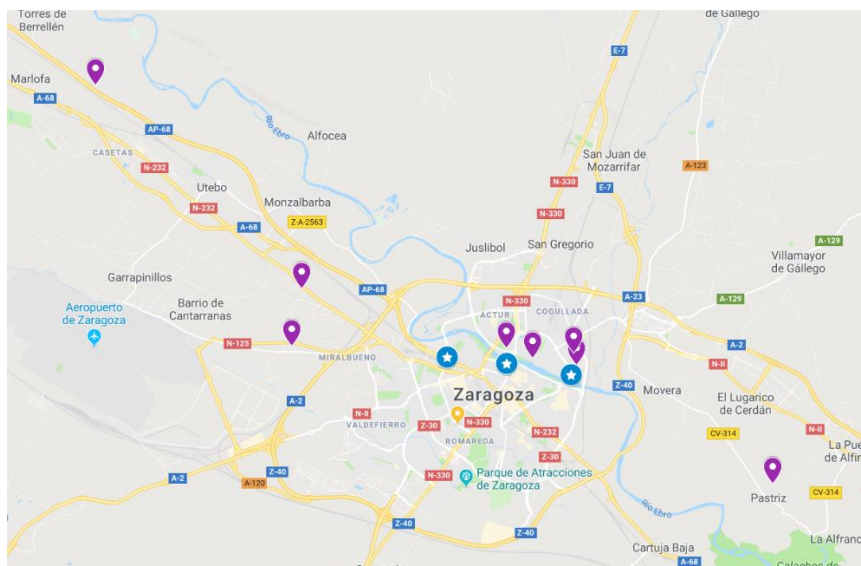
Todo este camino hasta la aplicación no se hace con los datos en plano. Los dispositivos no envían los mensajes a un destino, se transmiten los datos al aire, que es un medio inseguro, y cualquier receptor dentro del rango de alcance puede escucharlos. El dispositivo cifra el payload con una clave generada a nivel de aplicación TTN o app key. La aplicación TTN se encarga de descifrar el mensaje y aplica un control de integridad sobre el mensaje usando AES-128[4]. Gracias a esto se establece una comunicación segura extremo a extremo de los datos desde que el mensaje sale del dispositivo hasta que llega a la aplicación TTN tal y como se representa en la sección E.

Como se observa, TTN permite que la información de los dispositivos llegue a internet, sin embargo no la guarda, siendo necesaria la integración con otras aplicaciones.

El objetivo es estudiar las diferentes integraciones que ofrece la aplicación de TTN para diseñar una infraestructura, representada en la sección F, que gestione el almacenamiento de los datos, los dispositivos, el flujo de los mensajes y la visualización de los datos tras ser filtrados, permitiendo a los usuarios, representados en la sección G, puedan explotarlos.

## 2.2. Usuarios

Las casetas están repartidas por la zona de Zaragoza en los colegios participantes cuya localización se representan en la Figura 3 en color morado.



**Figura 3.** Mapa con los dispositivos del proyecto en morado y los gateways en azul

La herramienta integrada para los alumnos y profesores que permita visualizar los datos se debe caracterizar por su rápido aprendizaje y solo contendrá información del dispositivo o dispositivos asociados al colegio que pertenezcan. Sin embargo existen unas tareas de administración del proyecto que requerirá del control sobre todos los datos, la posibilidad de gestionar los dispositivos, así como la gestión del almacenamiento.

Debido a los objetivos tan dispares encontrados en el proyecto es conveniente separar los requisitos en función de los usuarios que explotaran las soluciones.

El proyecto va destinado a alumnos y profesores de educación primaria y secundaria, esto quiere decir que no tienen conocimientos avanzados de tecnología y que para muchos será el primer contacto con el mundo del IoT. Se estudiarán herramientas intuitivas y atractivas permitiendo que los niños se adapten a ella con facilidad. Sera necesario poner en marcha una herramienta de visualización que cumpla unos requisitos pensados para fomentar el uso de esta a los profesores y alumnos a los que se hará referencia cuando se hable de usuarios.

La cantidad de información que cada colegio va a generar es una oportunidad para aplicar inteligencia sobre el global de los datos recogidos y poder analizarlos a una escala mayor al área de cada colegio. La gestión de estos datos, almacenamiento, y visualización va destinado a los gestores del proyecto y perfiles con conocimientos tecnológicos avanzados.

### 2.3. Requisitos

Debido a que existen dos roles que se han desarrollado en el apartado anterior y que tienen objetivos diferentes en el proyecto podemos separar los requisitos en dos conjuntos, uno enfocado en los **usuarios** y otro enfocado a los **gestores**. Estos nombres los usaremos durante la memoria para referirnos a alumnos y profesores por un lado y a las personas encargadas de administrar el proyecto por otro.

#### Requisitos usuarios

ID	Descripción
RF1	Los alumnos deberán usar una herramienta de <b>visualización</b> de los datos de los dispositivos.
RF2	No debe suponer generación de código por parte del alumno.
RF3	Debe ser <b>accesible</b> desde cualquier punto.
RF4	Permitir programar alertas y eventos que envíen <b>notificaciones</b> .
RF5	Se deben poder crear, modificar y eliminar <b>paneles de control</b> o dashboards.
RF6	Los datos deben poder visualizarse en plano.
RF7	Los datos deben poder representarse en <b>graficas</b> .
RF8	Los datos deben poder ser <b>exportados</b> fuera de la herramienta de <b>visualización</b> para ser importado en programas como Excel.
RNF1	Las <b>notificaciones</b> pueden ir dirigidas a números de teléfono o a direcciones de correo.

**RNF2** La herramienta ha de ser **intuitiva y atractiva**.

#### Requisitos gestores

ID	Descripción
RF1	<b>Controlar el flujo</b> de los datos de todos los dispositivos.
RF2	<b>Almacenar</b> los datos de manera <b>persistente</b> .
RF3	Asegurar la <b>durabilidad</b> de los datos almacenados.
RF4	<b>Monitorizar</b> los dispositivos para comprobar su correcto funcionamiento.
RF5	<b>Visualización</b> de los datos almacenados.
RF6	La herramienta de monitorización y visualización debe ser <b>accesible</b> desde cualquier punto.
RF7	La herramienta de visualización para la gestión del proyecto deberá permitir generar usuarios de solo lectura para la aplicación en caso de que algún profesor o investigador desee el acceso a todos los datos.
RF8	La herramienta de visualización deberá permitir la creación de <b>alarmas y notificaciones</b> para saber el estado de los dispositivos.
RF9	<b>Gestionar</b> los dispositivos.
RNF1	Visualizar a simple vista que dispositivos están funcionando o no.

Aunque haya dos grupos de requisitos diferentes el estado actual del proyecto también establece un par de requisitos que deben tenerse en cuenta durante el desarrollo.

Este proyecto está impulsado por Ibercivis y Etopia, con presupuesto para la compra de material pero no para la gestión de la información, esto supone que se deberá de buscar el **mínimo coste** en la solución a implementar en explotación.

Por último, el hecho de poder visualizar los datos o administrar los dispositivos desde cualquier parte obliga a que el entorno a implementar sea público. Esto supone un reto de **seguridad** que conlleva el aislamiento al máximo del entorno para evitar que se puedan explotar vulnerabilidades en los diseños propuestos.

### 3. Análisis de elementos del diseño

En este bloque se expondrán las características que deben tener las diferentes herramientas o servicios, que en bloques posteriores se definirán, sobre las que basaremos los diseños a implementar. Todos los diseños se apoyarán en la base establecida por TTN e intentarán aprovechar las características de esta plataforma para el planteamiento de diseños sólidos.

En nuestro caso el número de dispositivos es actualmente bajo pero se espera que el proyecto evolucione. Para comenzar será necesario el uso de alguna herramienta que permita realizar la gestión **de los dispositivos**, crearlos, modificarlos e incluso eliminarlos teniendo en cuenta esa evolución del proyecto.

La plataforma de TTN solo gestiona el transporte de la información hasta la aplicación TTN. Con un cálculo realizado sobre los datos que llegan a la aplicación TTN, se ha obtenido el flujo de mensajes con los dispositivos actuales que deberá tenerse en cuenta durante el estudio de los diseños. El mensaje enviado por cada dispositivo contando con los datos de los sensores y los metadatos añadidos por TTN ocupa 820 Bytes. Cada hora un dispositivo manda 6 mensajes, para calcular el flujo de datos de n dispositivos se utiliza la siguiente formula:

$$820 \times n \times 6$$

En un escenario ideal donde no se pierde ninguna trama y para n=15 que es el número actual de dispositivos, se obtiene un flujo de 73.8 Kbytes/hora.

El diseño deberá contar con algún **servicio de almacenamiento persistente** que sea capaz de asumir el flujo de datos calculado antes, que asegure la durabilidad de los datos y proporcione algún método para realizar copias de seguridad que permitan la recuperación los datos en caso de fallo del sistema. Se realizará un estudio sobre los diferentes tipos de almacenamiento, en objetos (ficheros) o en tablas.

Para poder tener **control sobre los datos**, habrá que estudiar el uso de herramientas o aplicaciones que permitan desplegar código para gestionar el flujo de los mensajes, desde capturarlos, hasta aplicar filtros para enviar los datos personalizados a la herramienta de almacenamiento. Se estudiará la posibilidad de realizar un programa que permita esto en lenguajes de programación y servicios como Node-Red.

Por último, se deberá estudiar el uso de herramientas de **monitorización** que permitan la visualización de datos a través de graficas desde cualquier lugar. Así como la creación de cuadros de mando y de paneles de alarmas que notifiquen en caso de mal funcionamiento. Tras el estudio de las alternativas existentes, se decide reducir a 3 las herramientas sobre las que se plantearan los diseños:

- Cayenne, plataforma de visualización IoT con una interfaz muy intuitiva y que encajaría con los requisitos establecidos por el grupo escolar.[5]
- Grafana, software que permite una personalización muy amplia, integrándose fácilmente con herramientas de almacenamiento en tiempo real comúnmente usadas para proyectos IoT y que cumple con todos los requisitos de visualización del grupo de la gestión del proyecto.[6]
- AWS CloudWatch, es el panel de visualización de AWS permitiendo monitorizar datos de cualquier servicio desplegado en el Cloud.[7]

La herramienta tiene que ser de carácter público y se deberán usar servicios que aseguren la **integridad** del sistema y eviten ser objeto de vulnerabilidades básicas pero que comprometen la **seguridad** del entorno.

Tras un primer análisis los elementos que se han destacado son comunes en la mayoría de los proyectos de IoT.

Aunque durante el análisis de los requisitos se hayan diferenciado los requisitos de dos grupos, se buscaran diseños que satisfagan ambos.

Estos elementos identificados como características que deberán tener las herramientas a usar son comunes en un paradigma computacional llamado la "Cloud". La Cloud es un conjunto de servicios de computación ofrecidos a través de internet [8]. Este conjunto de servicios permitiría realizar una **gestión de los dispositivos** más ágil, pudiendo por ejemplo, automatizar la creación de ellos. La Cloud ofrece soluciones de **almacenamiento** muy variadas que podrían ser utilizadas en nuestro proyecto. Además está muy enfocada a servicios **públicos** lo que permitiría poder utilizar esas herramientas de **visualización** desde cualquier parte y de forma **segura**.

Las posibles integraciones en el Cloud con TTN son Amazon, Microsoft y Google entre otras. Como se ha comentado anteriormente el proyecto se comenzó en la asignatura de Laboratorio de Sistemas Empotrados (LAB SE), donde tras definir unos prerrequisitos se decidió usar AWS IoT para desarrollar nuevos conocimientos. Por ser el Cloud con más experiencia, disponer de servicios con los elementos descritos y dado que ya se había marcado el uso de un proveedor en LABSE, se decidió estudiar la Cloud de Amazon, Amazon Web Services (AWS)[9] al mismo tiempo que una arquitectura sobre un servidor administrado por ibercivis, para plantear diseños que permitan cumplir los objetivos y requisitos establecidos en la fase de análisis y valorar que solución es mejor para nuestro proyecto.



## 4. Estudio de los diseños

El objetivo de este estudio es valorar diseños que implementan diferentes tipos de soluciones para el proyecto y analizar concepto de complejidad, costes, plazos, potencial, seguridad y viabilidad entre otros.

Gracias a este estudio podremos evaluar diseños basados en AWS IoT[10], que es el servicio de AWS destinado a IoT que permite gestionar dispositivos, así como controlar el flujo de sus mensajes. Se evaluarán también diseños basados en integraciones con terceras aplicaciones y servicios no específicos de IoT en AWS.

Con el estudio, por último, de una solución basada en una instalación tradicional u on-premise, para decidir entre implementar una solución basada en Cloud o una alternativa más tradicional en una máquina virtual administrada por Ibercivis.

Para llevar a cabo este estudio, se planteará un modelo conceptual de cada diseño donde se explicará cada uno de los servicios y que objetivo cumplen dentro del diseño global. Seguidamente se configurará un entorno de demostración donde se implementarán los elementos mínimos que garanticen la viabilidad del producto. Esa demo servirá para poder evaluar los compromisos que deben cumplir las soluciones y sobre esa demo se realizara una evaluación para concluir cuál de los diseños es el más apropiado para el estado actual del proyecto.

A continuación se expondrán los modelos conceptuales, implementaciones y evaluaciones realizadas sobre los 4 diseños.

### 4.1. Integración TTN – AWS IoT

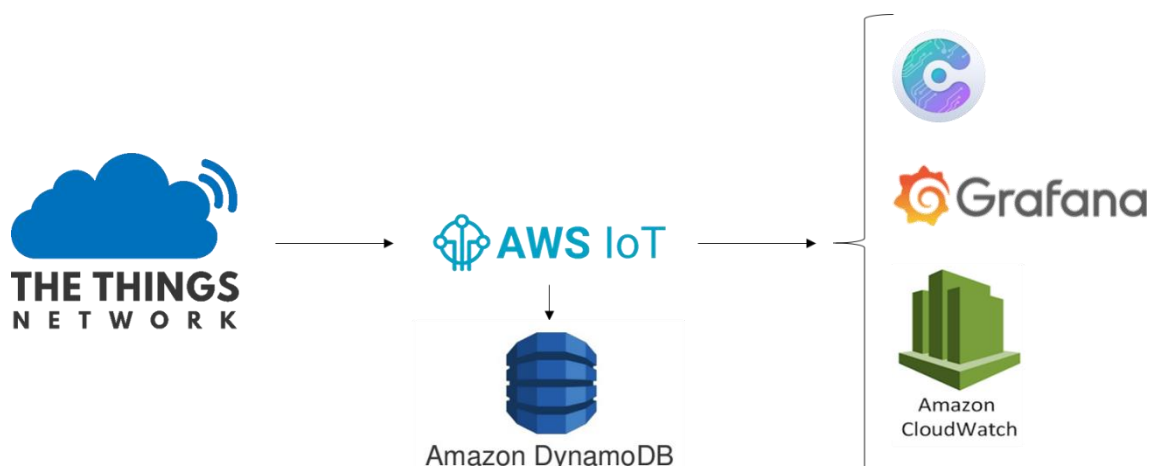
En primer lugar se va a estudiar el uso de servicios específicos de IoT dentro del Cloud como es AWS IoT, servicio que engloba soluciones de gestión de dispositivos, almacenamiento y análisis de datos para cumplir con los requisitos establecidos en el análisis del diseño.

#### 4.1.1. Diseño

El diseño representado en la Figura 4 basa su funcionamiento en la integración de TTN con el servicio de AWS IoT. Esta integración se expone en los medios oficiales de TTN como una integración que permite la sincronización de dispositivos, mensajes y registros o logs entre AWS IoT y TTN.

El uso de esta integración abre la posibilidad, en caso de escalado en el número de dispositivos, de automatizar la gestión de estos. Permitiendo administrar los dispositivos desde AWS IoT y hacer que TTN solo sea la plataforma que permite la comunicación de los datos desde los sensores hasta Internet. Esta implementación

facilitaría el uso de todos los servicios del ecosistema AWS que se necesiten incrementando el potencial del diseño.



**Figura 4.** Diseño basado en la integración TTN – AWS IoT

Se ha comentado antes la intención de estudiar el almacenamiento de la información en objetos o tablas. Este diseño plantea el uso de DynamoDB[11] como servicio de almacenamiento en tablas con un lenguaje NoSQL clave-valor que solucionaría la necesidad de almacenamiento persistente y replicado. Un evento programado cada vez que llegue un mensaje se encargara de llevar los datos de ese mensaje a las tablas de la BBDD accesibles desde cualquier parte.

Para la monitorización y visualización de los datos se plantea como primera opción el uso de AWS CloudWatch con el fin de evaluar la posibilidad de basar todo el diseño en servicios de AWS, pero el diseño es apto para la utilización de las 3 representadas en la margen derecha de la Figura 4.

#### 4.1.2. Pruebas

Se llevó a cabo la implementación de un entorno experimental con todos los elementos comentados anteriormente.

La instalación de la integración permitió comprobar como el registro de dispositivos en AWS IoT era replicado en TTN tras unos minutos de retraso.

Gracias al paso anterior los mensajes llegan hasta AWS IoT, se programaron eventos para guardar los datos en tablas de DynamoDB utilizando como clave la marca de tiempo o timestamp que venía incluido en el payload del mensaje.

En cuanto intentamos poner en marcha la monitorización y visualización de los datos sobre AWS CloudWatch observamos rápidamente que la herramienta no cumplía todos los requisitos establecidos, sobre todo para los colegios. La generación de las gráficas conllevaba programación, la interfaz no era la apropiada para un alumnado de primaria o ESO y los datos no se podían ver desde CloudWatch en crudo y por lo tanto tampoco se permitía exportarlos a otras herramientas como Excel.

Podría haberse solventado este requisito con la implementación de alguna solución pero los costes de tiempo comprometían la entrega del proyecto.

Se descarto esta herramienta para dar paso a Cayenne y cumplir los objetivos de visualización de los usuarios.

Debido al gran aislamiento que realiza AWS sobre sus servicios y la volatilidad del entorno sobre el que está desplegada la integración no fue posible mantener un flujo de datos constante hacia Cayenne.

#### 4.1.3. Conclusiones

A pesar de que el potencial del diseño es enorme debido a la posibilidad de usar todos los servicios de AWS, evaluando los compromisos marcados al inicio de este estudio se puede concluir que este diseño es demasiado complejo y el coste tanto en tiempo como en dinero escapa a la magnitud del proyecto.

El diseño no alcanza a cubrir los requisitos de visualización ni de gestión de los datos, además el hecho de tener que desarrollar un programa que mande los datos a las herramientas compromete la fecha de entrega del proyecto a los colegios.

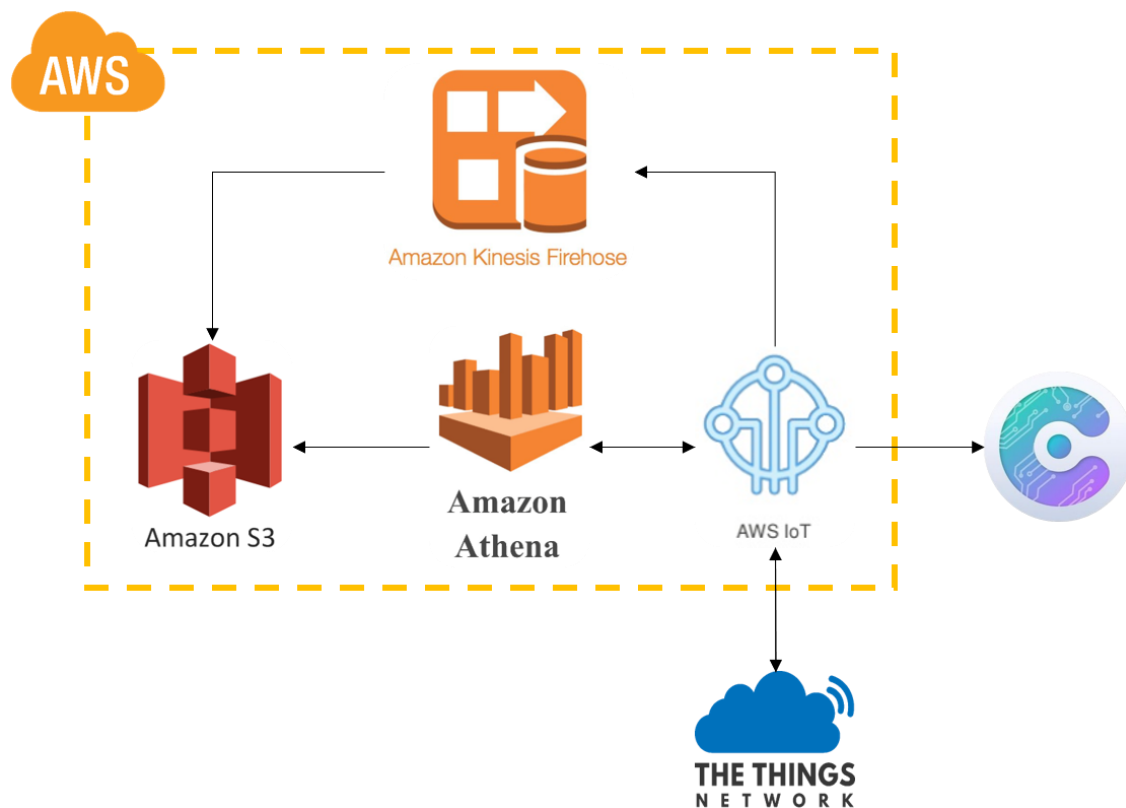
Los requisitos temporales y el nulo presupuesto hacen que no sea posible la implementación de esta infraestructura para el estado actual de este proyecto.

#### 4.2. Integración TTN – AWS IoT (Versión 2)

En segundo lugar se quiere estudiar el uso de servicios de AWS que apoyados en el servicio de IoT de la plataforma faciliten la llegada de los datos a la herramienta de visualización Cayenne. Además se busca estudiar el uso de servicios que realicen una gestión del almacenamiento basado en ficheros como alternativa al almacenamiento en tablas. El objetivo es valorar cuál de los dos modelos es más apropiado para este tipo de proyectos aun sabiendo que para el estado actual el diseño es demasiado complejo y supone demasiados costes.

##### 4.2.1. Diseño

Este diseño representado en la Figura 5 tiene como base la integración entre TTN y AWS IoT que ya se ha comentado antes y surge para plantear una alternativa de almacenamiento con S3[12], servicio de almacenamiento de AWS basado en objetos (ficheros), en vez de DynamoDB y evaluar ambas soluciones para un modelo de proyecto estándar de IoT.



**Figura 5.** Diseño basado en la integración + conjunto de servicios con almacenamiento S3

Gracias a la integración los datos llegan a AWS IoT donde a través de eventos programados se reenvían los mensajes a un flujo de Kinesis Firehose[13].

Kinesis Firehose es un servicio que permite cargar datos en tiempo real de manera fiable en almacenes de datos, es capaz de analizar los datos en tiempo real, y procesarlos por lotes, comprimirlos y cifrarlos con el fin de minimizar la cantidad de almacenamiento y aumentar la seguridad. Los mensajes se encolan en un flujo donde se establecen unos parámetros, en este caso el tamaño del fichero y el tiempo de espera. Con estos parámetros, los mensajes se van concatenando en un mismo fichero hasta que se cumple la condición de tamaño del fichero o la de tiempo de espera, entonces el fichero es enviado a S3 donde se queda almacenado como un objeto. Con esto se cumplen los objetivos de almacenamiento persistente y replicado, así como la posibilidad de acceder a los datos desde cualquier parte de manera segura. La integridad de los datos también estaría asegurada dado que el transporte de los mensajes hasta el almacenamiento queda aislado dentro de la plataforma de AWS.

En este diseño los datos están accesibles en ficheros, es necesario el uso de algún servicio que permita consultar datos sobre estos ficheros para llevar la información de manera selectiva a la herramienta de monitorización.

Amazon Athena[14] es un servicio de consultas interactivo que facilita el análisis de datos en S3 con SQL. A través de consultas SQL nos permite extraer información de ficheros almacenados en S3 como si de una BBDD se tratase habiendo antes definido la

estructura de los ficheros. Gracias a esto y de la AWS CLI[15], interfaz de comandos de AWS, se puede transportar la información deseada del servicio de almacenamiento al servicio de monitorización con la ejecución de un script.

#### 4.2.2. Pruebas

Para realizar las pruebas de este diseño desplegamos un entorno experimental donde configuramos los eventos para el reenvío de los mensajes a Kinesis, creamos y configuramos el flujo que encola los mensajes en ficheros y los almacena en S3

Se comprueba el correcto funcionamiento de Kinesis con S3. Tras haber analizado el tamaño ideal de los ficheros para su posterior consulta, se quiso forzar el límite de tiempo a un periodo normal de un día, debido a que los mensajes llegan cada 10 minutos, que solo son 15 dispositivos y que no todos están transmitiendo datos se cumple antes la condición de tiempo que la de tamaño y se agrupan los mensajes en ficheros cada día. Esto provoca que el número de ficheros a analizar en una consulta sea mayor al estimado, perdiendo eficiencia frente a la consulta sobre una tabla de DynamoDB.

Se instalo y configuro un cliente Mosquitto[16], que utiliza un protocolo de intercambio de mensajes llamado MQTT. Se estudio el uso de esta herramienta mediante la creación de un script que mandaba datos estáticos a Cayenne. Para comparar complejidad y coste de desarrollo de una solución basada en esta herramienta se realizó también un programa con la misma funcionalidad usando el lenguaje Nodejs[17]. Ambos programas atacan al punto de acceso o endpoint MQTT de Cayenne “mqtt.mydevices.com” con un usuario, contraseña e id proporcionados por Cayenne. Para que se pueda generar la gráfica se manda a un canal específico y se le indica la métrica o dato especificando el tipo y el valor.

El comando para poder mandar datos a Cayenne en la implementación basada en Mosquito se puede ver en la Figura 6.

```
mosquitto_pub -h <Endpoint> -i <DevID> -u <UserName> -P <Password>  
-t v1/<UserName>/things/<DevID>/data/<Channel>  
-m <DataType>,<shortDataType>=<Value>
```

**Figura 6.** Comando para mandar datos a Cayenne utilizando el cliente MQTT Mosquitto.

Las funciones utilizadas en la implementación de Nodejs pertenecen a una librería llamada CayenneLPP.

El servicio de Athena no se llegó a implementar debido a su compleja configuración y por saber de antemano que no cumplía los requisitos establecidos.

#### 4.2.3. Conclusiones

El diseño planteado es demasiado complejo para el problema concreto que se está planteando resolver, esta infraestructura en un entorno corporativo que tiene un presupuesto y con un volumen de la orden de miles de dispositivos cobraría sentido siendo una opción que permitiría una escalabilidad totalmente automática y cumpliría con todos los requisitos de visualización al poder crear tantas cuentas de Cayenne como se desee o enviar todos los datos a la misma cuenta.

La conclusión del estudio realizado para analizar qué tipo de almacenamiento es mejor para un proyecto de IoT, viene una vez más condicionada por la envergadura del mismo. Para el estado actual de nuestro proyecto, es una solución más barata y eficiente usar DynamoDB, sin embargo en el escenario presentado en el párrafo anterior, con una tasa de datos que permitiera el tamaño óptimo de los ficheros, la solución que guarda los datos en S3 permitiría ahorrar costes de almacenamiento sin sacrificar rendimiento en las consultas.

Tras el análisis de dos diseños que plantean el uso de AWS IoT, las implementaciones realizadas y los resultados obtenidos en ambos diseños, se concluye que la complejidad de los servicios junto al coste que suponen hace inviable el paso a producción de cualquiera de estos. Aunque los diseños no puedan implementarse en producción, la fase de estudio ha permitido extraer conclusiones interesantes sobre el tipo de almacenamiento a usar en proyectos IoT.

#### 4.3. TTN – AWS Free Tier

Este diseño plantea el estudio de servicios que aun siendo de AWS se ajusten al presupuesto y el número de dispositivos actual, aunque esto suponga perder otros que permitan la gestión de los dispositivos o que reduzca el potencial del diseño al perder la integración con determinados servicios.

Se plantea estudiar un diseño que realice el transporte de los datos a Cayenne para cumplir con los objetivos de los usuarios, con los tiempos del proyecto y con el presupuesto actual dentro de la Cloud de AWS.

##### 4.3.1. Diseño

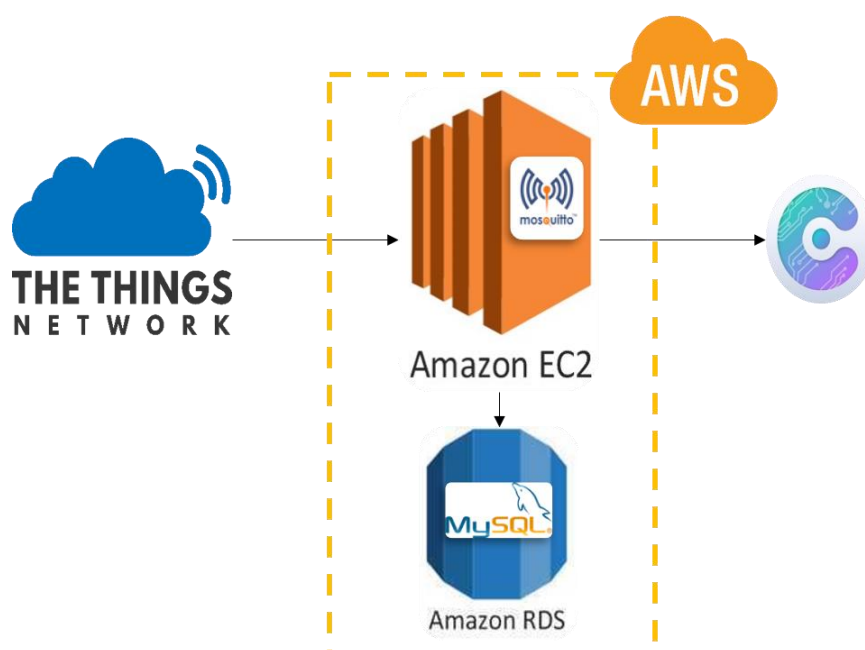
Amazon EC2[18], representado en la parte central superior de la Figura 7, es el servicio que proporciona capacidad informática en la nube de forma segura y de tamaño modificable. Permite levantar máquinas virtuales o instancias, con unas características muy personalizables, desde el tipo de instancia que define el número de núcleos que tendrá, a un Security Group[19] que se asocia a una instancia y que sirve para controlar el acceso a los puertos de la misma, pasando por las opciones de disco donde podemos

elegir tamaños y tipos entre otras opciones de personalización de la instancia. Existe una gestión automática de copias de seguridad por parte de AWS que realiza una AMI (Amazon Machine Image)[20] todas las noches permitiendo recuperar los datos del día anterior en caso de fallo.

El uso de instancias de tamaño t2.micro (1 vCPUs, 2.5 GHz, Intel Xeon Family, 1 GiB memory) entra dentro de la capa gratuita con la condición de no superar las 750 horas de cómputo total en un mes. Esta condición nos permite tener una instancia con IP publica encendida todos los días del mes.

Amazon RDS, representado en la parte inferior de la Figura 7, es el servicio que gestiona bases de datos relacionales, al igual que con las instancias se pueden configurar múltiples parámetros de los cuales muchos comparten como el tamaño, los Security Groups. RDS es un servicio general que permite instalar en tu BBDD el motor deseado entre varias opciones (AuroraDB, PostgreSQL, MySQL, MariaDB, Oracle y SQL Server). AWS se encarga al igual que con las instancias de realizar snapshots automáticos todos los días para que la pérdida de datos sea mínima en caso de fallo.

El uso de RDS de tamaño db.t2.micro (1 vCPU y 1GiB RAM) con 20GB de espacio para la BBDD y otros 20 GB para copias de seguridad de BBDD y snapshots de BBDD. Sabiendo el flujo de datos en Bytes/hora que se ha calculado en el análisis del diseño se puede saber aproximadamente cuanto durarían los 20 GB en caso de que los 15 dispositivos estuvieran transmitiendo 24x7 sin perder ninguna trama. En un día cada dispositivo manda 144 mensajes, el espacio consumido en un día por todos los dispositivos seria de 1,77MB, lo que hacen un total de 30 años aproximadamente hasta que se llenara el espacio de la capa gratuita. A esta RDS se le instalaría el motor de BBDD Mysql ya que es el único motor dentro de la capa gratuita.



**Figura 7.** Diseño en busca del coste 0

La gestión de los dispositivos se quedaría dentro de la consola de TTN, lo cual no supondría un problema debido al bajo número de dispositivos y que no se realizan altas y bajas de forma continua. El almacenamiento quedaría solventado con la RDS y la durabilidad con las copias de seguridad automáticas tanto de la RDS como de la instancia EC2, faltaría entonces cumplir los requisitos de la herramienta de visualización en la que Cayenne sigue siendo la opción para implementar.

Dado que la capa gratuita permitiría implementar el diseño sin coste alguno cumpliendo los compromisos del estudio, vamos a ver como se cumplirían los requisitos establecidos por los usuarios y gestores.

#### 4.3.2. Pruebas

En un entorno experimental se configuro y despliego una instancia EC2 con las especificaciones indicadas anteriormente, a través de ella lo que se quería conseguir era un flujo de datos en tiempo real de TTN a Cayenne que pasara a través de la instancia para poder aplicar inteligencia a los datos y sacar estadísticas de los datos actuales comparados con los de la RDS o almacenarlos en esta.

Gracias al cliente MQTT Mosquitto, con el que antes habíamos conseguido publicar datos a Cayenne utilizando un tema o topic específico, se consiguió también suscribirse al topic de TTN donde se publicaban los mensajes que llegaban de nuestros dispositivos con el comando de la Figura 8. Con el comando podíamos dejar un proceso en background o incluso convertirlo en servicio y conseguir que todos los mensajes llegaran a la instancia EC2, desde ahí, podíamos insertarlos en la RDS y mandarlos a Cayenne.

```
mosquitto_sub -h eu.thethings.network -u <UserName> -P <Password> -t <Topic> -v
```

**Figura 8.** Comando para publicar datos en un topic de TTN

En la documentación de Cayenne, se plantean diferentes tipos de lenguaje para mandar los datos, entre ellos NodeJS. Se consiguió conectarse con un programa al endpoint de Cayenne y publicar algún dato.

Para que el flujo de datos fuera continuo debía llevarse a cabo una implementación más elaborada que comprometía la fecha de entrega del proyecto siendo esta uno de los compromisos del estudio. Es por eso que no se realizó la configuración ni el despliegue de la RDS, porque no influía en el resultado del estudio del diseño.

#### 4.3.3. Conclusiones

Tras el estudio del diseño para conseguir un coste 0, se puede destacar que existen servicios que permiten implementar soluciones en producción ajustando el coste al presupuesto actual. El proceso para implementar el diseño es más laborioso y eso



provoca que no se pueda aplicar esta solución a nuestro proyecto por cuestiones temporales.

#### 4.4. Conclusiones AWS

Tras realizar un estudio de 3 diseños basados en la Cloud de AWS es necesario exponer las conclusiones comunes que han aparecido durante la evaluación de estos diseños.

Los diseños basados en el servicio de IoT de AWS, se descartaron porque el coste y la complejidad de configuración que conllevaba implementarlos no lo justificaban 15 dispositivos mandando mensajes cada 10 minutos. La solución basada en servicios estándar de AWS se descartó porque su implementación sobrepasaba el límite para entregar el proyecto a los usuarios escolares aun habiendo cumplido los requisitos de coste y visualización.

La conclusión de estos 3 diseños en conjunto es que el servicio de AWS IoT ofrece una agilidad y una sencillez de integración que permite desplegar diseños relativamente complejos de manera rápida que para cumplir con los objetivos del proyecto a cambio de pagar por esos servicios. En caso de que la restricción sea monetaria en vez de temporal, el diseño de coste 0 cumple con todos los requisitos del proyecto tanto de los usuarios como de gestores.

En el caso de este proyecto ambos, el tiempo y el dinero, eran requisitos para la implementación de una solución, lo que lleva descartar del diseño a implementar el componente de la Cloud para valorar sustituirlo por una máquina virtual tradicional sobre un host administrado por Ibercivis.

#### 4.5. Diseño final

El estudio de la implementación final sobre una máquina virtual tradicional se lleva a cabo casi de forma imperativa por Ibercivis para tener el control sobre la infraestructura a desplegar, proporcionando eso si todos los elementos necesarios para asegurar la accesibilidad, la durabilidad y la seguridad de los datos.

Por un lado, se trata de llevar los datos a Cayenne a través de una integración directa entre TTN y Cayenne, representada en la parte derecha de la Figura 9, evitando tener que generar código y agilizando el proceso. De esta manera los colegios dispondrían de una herramienta de visualización que cumpla con los requisitos temporales de los usuarios. Esta solución había sido inicialmente descartada por duplicar el almacenamiento.

Por otro lado, estudiar un conjunto de servicios que permitan cumplir con los requisitos de los gestores, para los cuales no existen requisitos temporales fuertes. Este conjunto de servicios lo llamaremos a partir de ahora pila de servicios o Stack IoT,

representada en la parte izquierda de la Figura 9 dado que se estudiará una solución general para almacenamiento, gestión y visualización de datos IoT.



**Figura 9.** Diseño final

#### 4.5.1. TTN – Cayenne (Requisitos Usuarios)

Cayenne es una herramienta que permite de manera sencilla crear y personalizar dashboards, crear eventos y ver datos en crudo o con graficas. Tiene disponible tanto aplicación web como móvil de manera gratuita, lo que fomentaría el uso por parte de los alumnos de la herramienta.

Cayenne además guarda los datos que van llegando a la aplicación en una BBDD propia sin coste, esta característica hace que los datos vayan a replicarse porque de cara a la gestión del proyecto hará falta almacenar los datos en una BBDD administrada por nosotros.

La comunicación entre Cayenne y TTN se hace a través de una integración ya disponible, que consiste en compartir la clave de aplicación TTN, que permite en reenvío de los datos de TTN al endpoint de Cayenne. Así con registrar el ID del dispositivo en la herramienta de Cayenne se recogerán los datos de ese dispositivo o caseta.

##### 4.5.1.1. Formato de la trama

Como se explica en la introducción, los datos se mandan con un formato que en el envío se codifica en lo que llamamos carga útil o payload y que en el destino se decodifica generando un mensaje en claro, Figura 13. A la hora de elegir un formato para codificar las tramas se analizaron dos opciones.

La primera, establecer un formato personalizado para los datos que generan los dispositivos de manera que ocupen lo menos posible. En caso de elegir esta opción se deberían haber generado dos librerías, una que se encontraría dentro el dispositivo y se encargaría de codificar los datos según el tipo y otra que se tendría que subir a TTN

para que el servidor de aplicación fuera capaz de decodificar la trama que le está llegando y poder tener los datos en claro.

La segunda, elegir un formato ya existente con unas librerías que permiten la codificación sencilla de la trama y que ya establecen un formato determinado para cada tipo de dato.

IPSO Alliance, era una organización que promovía la estandarización de la representación de los tipos de datos a través del uso de un “Object ID” para cada tipo.

Este es el caso de CayenneLPP[21], formato de datos que permite enviar en la misma trama datos de diferentes sensores y que cumple con las restricciones de tamaño que establece LoRa. Para reducirlo, CayenneLPP también permite mandar datos del mismo dispositivo en diferentes tramas especificando el dispositivo al que pertenecen en el topic.

CayenneLPP identifica cada tipo de dato con un “Object ID”. Debido a que el tamaño de las tramas es importante se realiza una conversión para ajustar el ID de los objetos en un solo byte.

`LPP_DATA_TYPE = IPSO_OBJECT_ID - 3200`

Cada tipo puede usar 1 o más bytes para mandar los datos oportunos tal y como se indica en la Figura 10.

Type	IPSO	LPP	Hex	Data Size	Data Resolution per bit
Digital Input	3200	0	0	1	1
Digital Output	3201	1	1	1	1
Analog Input	3202	2	2	2	0.01 Signed
Analog Output	3203	3	3	2	0.01 Signed
Illuminance Sensor	3301	101	65	2	1 Lux Unsigned MSB
Presence Sensor	3302	102	66	1	1
Temperature Sensor	3303	103	67	2	0.1 °C Signed MSB
Humidity Sensor	3304	104	68	1	0.5 % Unsigned
Accelerometer	3313	113	71	6	0.001 G Signed MSB per axis
Barometer	3315	115	73	2	0.1 hPa Unsigned MSB
Gyrometer	3334	134	86	6	0.01 °/s Signed MSB per axis
GPS Location	3336	136	88	9	Latitude : 0.0001 ° Signed MSB
					Longitude : 0.0001 ° Signed MSB
					Altitude : 0.01 meter Signed MSB

**Figura 10.** Tabla que representa los diferentes tipos de datos del formato CayenneLPP, en orden: Type (Tipo de dato), IPSO(ID que tiene en la IPSO Alliance), LPP (ID de 1 byte del formato CayenneLPP),

Hex(Representación hexadecimal del ID), Data Size(Tamaño del tipo de dato en bytes), Data Resolution per-bit(Precisión del dato)

Siguiendo la tabla anterior, el payload en hexadecimal para enviar los datos de dos temperaturas se refleja en la Figura 11.

Payload (Hex)	03 67 01 10 05 67 00 FF	
Data Channel	Type	Value
03 ⇒ 3	67 ⇒ Temperature	0110 = 272 ⇒ 27.2°C
05 ⇒ 5	67 ⇒ Temperature	00FF = 255 ⇒ 25.5°C

**Figura 11.** Formación del payload con CayenneLPP para dos temperaturas.

Estos cálculos son realizados para Arduino, por funciones de codificación que se encuentran en una librería llamada Cayenne LPP. Solo se debe indicar el canal por donde se va a mandar el dato y el valor. Algunas de estas funciones se presentan en la Figura 12.

```
uint8_t addPresence(uint8_t channel, uint8_t value);
uint8_t addTemperature(uint8_t channel, float celsius);
uint8_t addRelativeHumidity(uint8_t channel, float rh);
```

**Figura 12.** Funciones para codificar datos de la librería CayenneLPP

Se optó por elegir CayenneLPP debido a que es un formato didáctico para el profesorado más tecnológico e integrado en aplicaciones como TTN o Cayenne. Se comprobó que los mensajes se decodificaban en TTN de manera correcta obteniendo los datos en crudo en formato JSON como se indica en la Figura 13.

```
{
  "app_id": "pajaros_en_la_nube",
  "dev_id": "pajaro_ttnzg_008",
  "hardware_serial": "00954D6256DBA30E",
  "port": 1, "counter": 0, "is_retry": true,
  "payload_raw": "AWcASgJzJkwDaEoEZWDjBXmMwWZoSwdmAAgCAAAJAgJyCgAP",
  "payload_fields": {
    "analog_in_8": 0, "analog_in_9": 6.26,
    "barometric_pressure_2": 980.4, "barometric_pressure_5": 981.9,
    "digital_in_10": 15, "presence_7": 0,
    "relative_humidity_3": 37, "relative_humidity_6": 37.5,
    "temperature_1": 23, "temperature_4": 22.7
  },
  "metadata": {
    "time": "2019-06-04T21:12:39.521216904Z", "frequency": 868.1,
    "modulation": "LORA", "data_rate": "SF7BW125",
    "airtime": 97536000, "coding_rate": "4/5",
    "gateways": [
      {
        "gtw_id": "eui-30aea4ffffe89a8", "timestamp": 1634667648,
        "time": "", "channel": 0, "rssi": -62, "snr": 10, "rf_chain": 0,
        "latitude": 41.66286, "longitude": 0.96765, "altitude": 240
      }
    ]
  }
}
```

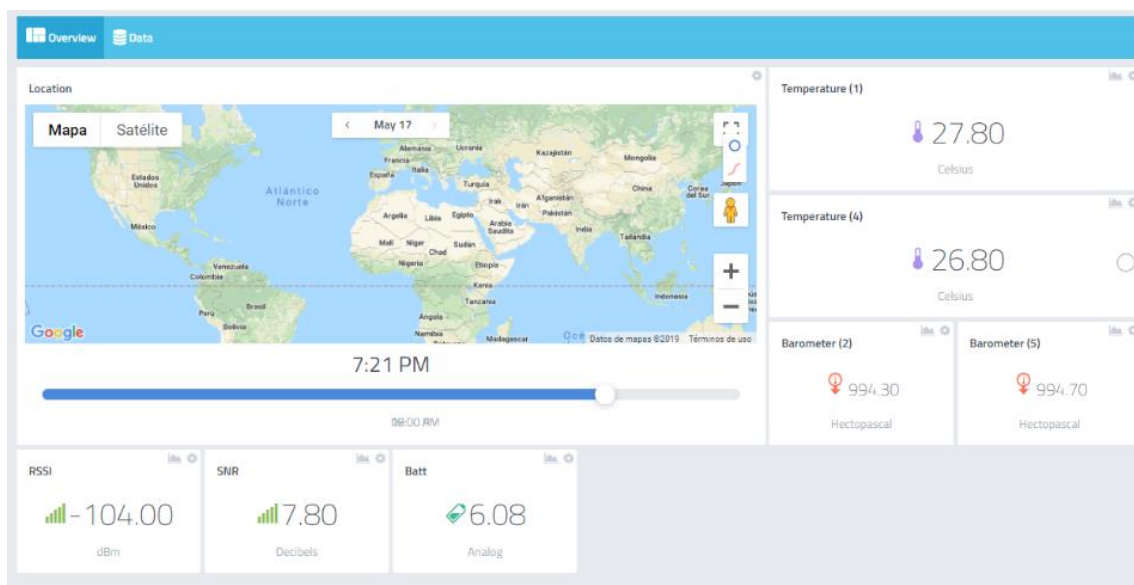
**Figura 13.** Mensaje decodificado en TTN

En el momento en que llegue el primer mensaje con los datos, se generaran los dashboards de manera casi automática, teniendo que cambiar el valor por defecto de alguno, gracias a que se especifican los tipos de datos, tal y como se representa en la Figura 13, en la trama al codificar los mensajes con CayenneLPP.

#### 4.5.1.2. Pruebas

Se instalo la integración en la aplicación de TTN donde están registrados los dispositivos y se creó una cuenta de prueba en Cayenne para registrar el dispositivo.

Como se ha indicado anteriormente con la llegada del primer mensaje se crearon automáticamente los dashboards, que se pueden ver en la Figura 14, caracterizados por ser intuitivos gracias al uso de iconos familiares y por tener una interfaz atractiva. En el panel se puede observar las diferentes posibilidades que ofrece como el uso de mapas geográficos para localizar el dispositivo y más herramientas que están explicadas en detalle en el manual anexo en el capítulo 8.



**Figura 14.** Panel de control de Cayenne.

#### 4.5.1.3. Conclusiones

La integración permite entregar el proyecto en tiempo estimado y la herramienta de Cayenne cumple con todos los requisitos establecidos por los Colegios.

Se decide entonces implementar esta solución para cumplir con las necesidades del proyecto escolar.

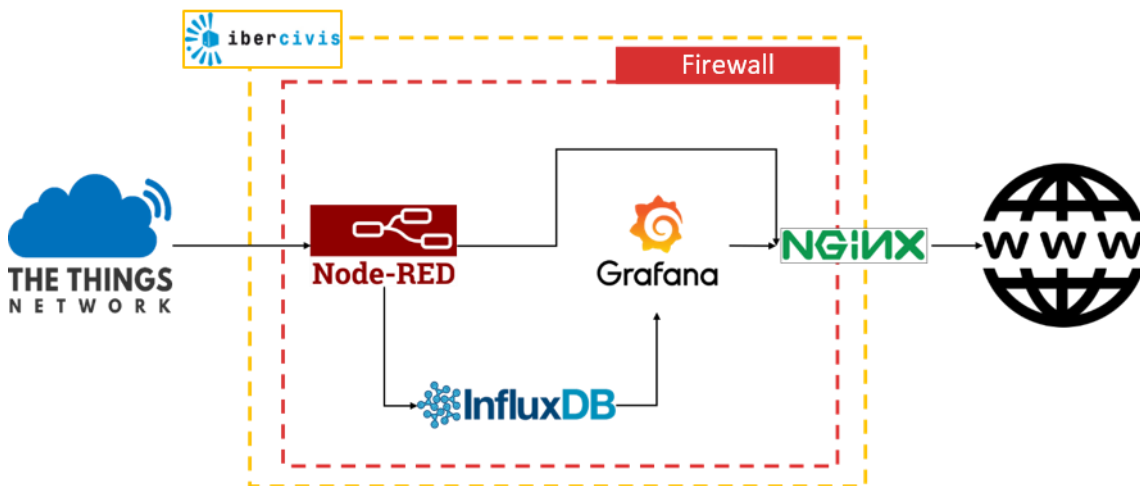
#### 4.5.2. Stack IoT (Requisitos de los gestores)

Para analizar el diseño de la stack se analizan herramientas que permitan cumplir los requisitos de los gestores, estudiando el uso de servicios gratuitos sobre un entorno Linux.

En la Figura 15 observamos que el conjunto de servicios que se plantean desplegar se enmarca dentro de dos cajas, la más interna que representa un cortafuegos o firewall y la externa que representa el host de Ibercivis donde esta alojada nuestra máquina. El servicio que las corta (NGINX), es el que representa el punto de acceso a la stack y que se encargara de realizar las funciones que fortalecer la seguridad y la integridad de los datos.

Hay que aclarar que la Figura 15 representa el diseño de producción ya que para esta fase de análisis se utilizó una instancia EC2 en vez de la maquina administrada por Ibercivis para agilizar las pruebas.

##### 4.5.2.1. Diseño



**Figura 15.** Diseño de la Stack IoT de producción.

Para desplegar código que permita el transporte de los mensajes desde TTN, de manera personalizada, hasta el servicio de almacenamiento es necesario estudiar el uso de herramientas como NodeRed debido a la intuitiva forma que tiene de desplegar programas.

NodeRed[22] es un motor de despliegues de flujos que permite definir gráficamente flujos de servicios que utilizan protocolos como MQTT o WebSocket. Es una herramienta visual programada en NodeJS que ofrece una sencilla interfaz HTML donde creamos un flujo con nodos. Este flujo se despliega sobre la maquina y se almacena en un fichero JSON que permite que en caso de reinicio de la maquina el flujo se despliegue solo.

Para almacenar los datos de manera persistente, organizar de forma automática y consultar de manera óptima la información basada en series de tiempo o timestamps se estudia el uso de servicios especializados en IoT como InfluxDB

InfluxDB[23] es un servidor de base de datos organizado en time series, ideal para logs o datos que se generan en vivo como es nuestro caso. InfluxDB permite generar columnas dinámicamente sin necesidad de fijar un esquema previo. En influxDB se pueden insertar dos tipos de datos, valores como tal que no son indexables y tags que si lo son permitiendo acceder a los datos de manera más selectiva. Además esta herramienta utiliza un lenguaje para acceder a los datos muy similar a SQL.

Para gestionar la monitorización de los dispositivos y la visualización de los datos almacenados en InfluxDB, la herramienta entre las 3 posibles que más se ajusta a los requisitos de los gestores es Grafana.

Grafana es una herramienta de visualización de datos basados en timestamps. Permite generar gráficos personalizados sobre datos que pueden obtenerse de diferentes fuentes, así como paneles de control, alarmas y notificaciones. Tiene integración con BBDD Mysql, PostgreSQL, y entre otras más, InfluxDB, y destaca por su versatilidad y potencia.

Por seguridad se quiere ocultar la existencia de estos servicios antes comentados dentro de la máquina virtual y evitar ser víctimas de ataques automatizados. Se analizará el uso de herramientas como Nginx[24] que permiten hacer de capa visible en una implementación con IP publica y segmentar los accesos a los servicios de la máquina de forma segura.

NGINX es un servidor web de alto rendimiento que también puede realizar funciones de proxy. Este servidor permite que todo tráfico web salga de nuestra maquina a través de él. Es la evolución generacional de Apache.

El firewall permite filtrar los accesos a la instancia a través de la apertura de puertos. Por defecto todos los puertos están cerrados y hay que abrirlos explícitamente.

El conjunto de todos estos servicios permite desplegar un flujo desde NodeRED que escucha los mensajes del bróker de TTN y los inserta en la BBDD InfluxDB aplicando filtros en el flujo para clasificar los datos como queramos. Desde Grafana se recolectan los datos y se configuran las gráficas y consultas SQL que permiten que los datos se muestren en vivo, además permite generar alarmas para saber si los dispositivos están funcionando bien y tener un panel de control que permita de un solo vistazo saber que dispositivos están vivos y cuáles no. Tanto la parte de despliegues de NodeRed, como la configuración y explotación de Grafana se hace a través de sus interfaces web, lo cual supone abrir una instancia con IP publica al mundo. A través de Nginx podremos restringir el acceso a unas URLs determinadas gracias a su función de proxy y añadir autenticación para securizar aún más el entorno.

Quedan abiertos al publico en este diseño el puerto 22 para la conexión por SSH y el 80 para la web.

De esta manera se cumplirían los requisitos establecidos por la gestión del proyecto de visualización, almacenamiento de datos, gestión de los dispositivos y control del flujo de datos.

#### 4.5.2.2. Pruebas

Tras el estudio de cada herramienta se instaló como primer servicio Nginx para mantener oculto en todo momento los servicios, que se iban implementando, a través de un proxy en el puerto 80.

A continuación, el servicio de NodeRed, que permitieron comenzar a realizar flujos que conectaran al bróker de TTN y añadiendo nodos para personalizar los datos que más tarde se insertarían en InfluxDB.

Se instalo después InfluxDB y tras la comprensión de su funcionamiento se añadió un nodo al flujo que realizaba la inserción de los datos en la tabla indicada.

Una vez que los datos se almacenaban correctamente se instaló Grafana donde a través de consultas SQL se generaron graficas para cada uno de los tipos de datos que recogían los sensores.

#### 4.5.2.3. Conclusiones

Tras la generación de los gráficos en Grafana se concluyó que el diseño propuesto cumplía con los requisitos de la gestión del proyecto, con coste 0, gracias al uso de servicios de código abierto, y permitiendo securizar el entorno para evitar intrusiones indeseadas.

Como se ha comentado, con el fin de agilizar la fase de pruebas, el entorno de demostración, con la instalación y configuración de todos los servicios, se realizo sobre una instancia EC2 de AWS y para el paso a producción se llevará a la maquina administrada por Ibercivis.



## 5. Implementación

Tras el estudio de los diseños finales y llegar a unos planteamientos viables que cumplen con todos los requisitos se va a desarrollar la implementación de los mismos. Primero con el diseño que cumple con los requisitos de los usuarios y después con el diseño que cumple los requisitos de los gestores.

### 5.1. TTN - Cayenne

Los requisitos escolares se basan en la implementación de una herramienta de visualización fácil y atractiva para que los alumnos y profesores se adapten sin problemas a ella y sea una herramienta de apoyo y no una herramienta que ralentice los objetivos escolares como el estudio de la fauna insectívora y de su hábitat. Además se deben cumplir los compromisos adquiridos durante el estudio de seguridad y coste entre otros.

Para llevar a cabo el paso a producción de una forma ordenada y sin arrastrar posibles confusiones de las fases de estudio, se procede a homogeneizar los nombres de los dispositivos bajo una aplicación TTN nueva.

A continuación realizamos la integración de esa aplicación nueva con Cayenne.

Con el fin de organizar los dispositivos y que no hubiera confusión en los datos a analizar por los usuarios, decidimos separar los dispositivos por colegios y que cada uno tuviera una cuenta de Cayenne.

El control sobre las cuentas de Cayenne es tarea del gestor, para ello generamos una dirección de Gmail por dispositivo con la que después creamos una cuenta de Cayenne que es la que finalmente les proporcionamos a los colegios. De esta manera las contraseñas de Gmail y de Cayenne las administramos nosotros y será tarea del gestor el resetear la contraseña en caso de olvido. Los nombres de las cuentas siguen un estándar [pajaroX.ttnzg@gmail.com](mailto:pajaroX.ttnzg@gmail.com) siendo X el número de dispositivo que este asociado a ese email. Por ejemplo, [pajaro1.ttnzg@gmail.com](mailto:pajaro1.ttnzg@gmail.com) correspondería con el correo asociado al dispositivo con el Device\_ID pájaro\_ttnzg\_001 en la aplicación de TTN. Las cuentas además se dejaron en un estado virgen donde Cayenne te obliga a que la primera acción de los colegios al entrar sea registrar un dispositivo.

Todo esto es con el fin de que sea el gestor y no el colegio el que tenga el control sobre todos los elementos necesarios para que los datos lleguen de manera correcta y ordenada a los usuarios.

Debido al perfil al que va enfocado este diseño se decidió crear un manual donde se explica paso por paso como crear un dispositivo. Tras esto se explican los diferentes elementos que tenemos disponibles en Cayenne para interactuar y como utilizarlos. El

manual incluye información tanto de la aplicación web como de la aplicación móvil y se adjunta como anexo.

Como paso final, se proporciona a los colegios la información necesaria, ID del dispositivo y credenciales de Cayenne, para poner en marcha la infraestructura desplegada por nosotros.

Respecto a los compromisos comentados antes, la herramienta se publicita como gratuita pero no conocemos la evolución que puede tener y la seguridad queda cubierta gracias a que el flujo de datos es administrado por la integración.

## 5.2. Stack IoT

Tras el estudio de las herramientas que implementarán la solución que permitirá, a los gestores, controlar los dispositivos, el almacenamiento, el flujo de los datos y la seguridad del entorno, se presentan los recursos proporcionados por Ibercivis para el paso a producción.

Ibercivis con el fin de tener control y reducir costes adicionales sobre toda la infraestructura desplegada, nos proporciona una máquina virtual Ubuntu 18. Esta máquina estaba dotada por una IP publica de su pool. Además debido a que Ibercivis es propietario del multidominio \*.ibercivis.es solicitamos la creación de un subdominio que no supone coste alguno del tipo ttnzgapi.ibercivis.es para que responda por la IP publica de la máquina virtual. Además Ibercivis comparte el certificado, con entidad certificadora externa, que tienen sobre el multidominio mencionado antes para poder cifrar las comunicaciones y que todo el tráfico publico vaya sobre HTTPS aumentando la seguridad y asegurando la integridad de las peticiones que llegaran a la máquina virtual.

Tras actualizar los repositorios y el kernel de la máquina se instalan los servicios en un orden que permita mantener la seguridad del entorno en todo momento y que no produzca dependencias entre los servicios que se van a integrar.

A continuación se describe la instalación, configuración y validación de cada servicio en el orden lógico que sigue el mensaje. No debe interpretarse este orden como un manual de instalación.

Para ello comenzamos por instalar el primero de los elementos de la stack, Nginx.

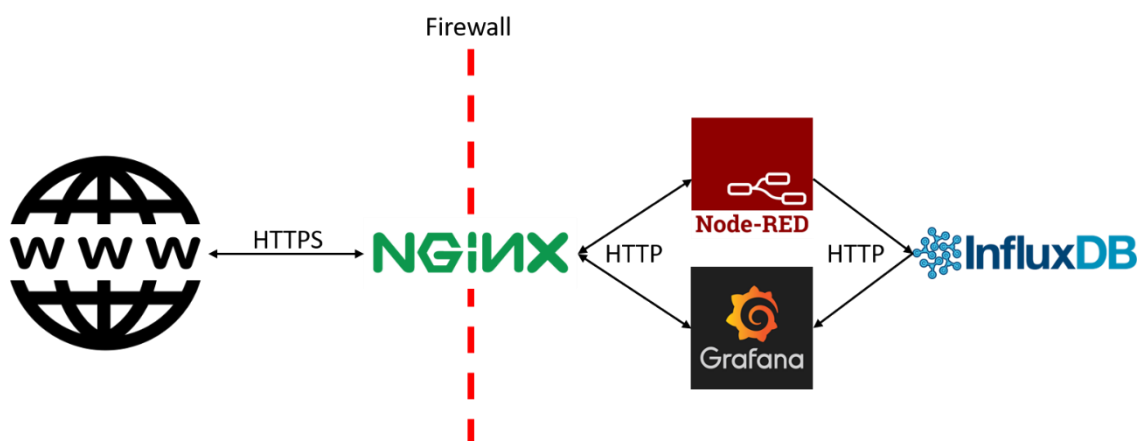
Aunque se ha mencionado levemente durante la explicación del diseño, es necesario comentar el propósito de este servicio dentro de la stack.

Cuando hablamos de máquinas virtuales que tienen IP publica, la seguridad es el primer requisito a tener en cuenta. El hecho de tener un entorno con una IP publica supone estar expuesto a ataques, estos ataques no son premeditados ni mucho menos directos hacia nuestro entorno, son escaneos aleatorios y automatizados que recorren

rangos de IPs públicas en busca de puertos abiertos a través de los cuales explotar vulnerabilidades.

Para evitarlo utilizamos un proxy que es un servidor que hace de intermediario entre un cliente y otro servidor. Gracias a esta función podemos conseguir aislar los dos servicios que requieren de acceso web y dejar visible solamente el puerto 80 en la máquina virtual. Una vez comprobado el correcto funcionamiento instalaremos los certificados para que el acceso sea a través de un puerto seguro como es el 443.

Es necesario comentar que en el esquema de comunicaciones descrito en la Figura 16, aparece solo como protocolo de entrada HTTPS, esto es el esquema final ya que para poder aplicar la certificación es necesario que las comunicaciones funcionen primero sobre HTTP.



**Figura 16.** Esquema de comunicaciones entre servicios y visibilidad externa.

Configuramos, un virtual Host en Nginx donde se le dice que escuche por el puerto 80 peticiones al dominio `ttnzgapi.ibercivis.es`.

Los servicios que vamos a integrar escuchan por 3 puertos diferentes.

Vamos a realizar la instalación y configuración de cada uno de ellos, en el orden lógico de los datos, de Node-Red a InfluxDB y de este a Grafana.

Tras la instalación de las dependencias de Node-RED se instala la aplicación y pasamos a configurarla. La herramienta es accesible desde el puerto 1880 y está escuchando por la interfaz pública, para aislar el servicio cambiamos la dirección por `localhost:1880` para que no pueda ser accesible directamente desde el exterior tal y como se representa en la Figura 16.

Debido a que estamos en un entorno en el que se busca la mínima necesidad de administración es necesario convertir la aplicación en un servicio administrado por `systemd`[25] que permite el inicio automático de la aplicación y su ejecución en el background. SystemD es un conjunto de bloques básicos de compilación para sistemas

Linux. Proporciona un gestor de sistemas y de servicios que se ejecuta con PID 1 y que se encarga de iniciar el resto del sistema entre otras tareas.

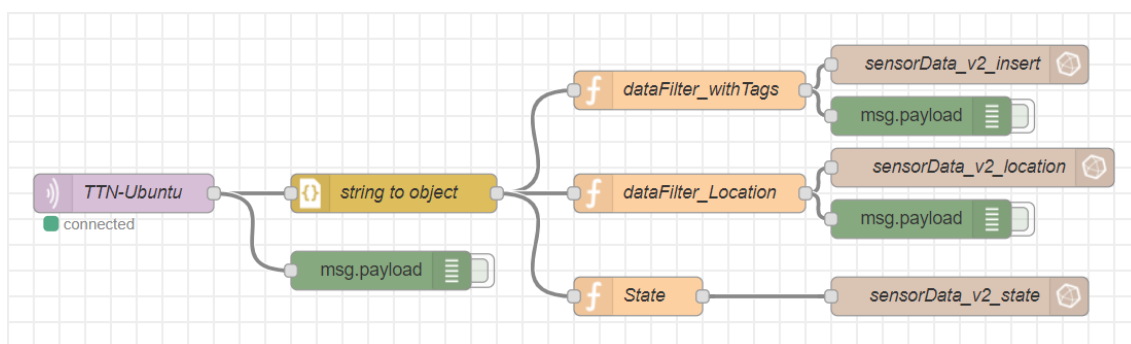
Una vez la configuración es correcta, procedemos a dar acceso a través de Nginx a la aplicación web.

Para hacer llegar las peticiones al servicio de Node-Red especificamos la siguiente norma en la configuración de `ttnzgapi.ibercivis.es`.

En caso de que llegue una petición con la URI `/nodered`, es decir <http://ttnzgapi.ibercivis.es/nodered> hacemos un reverse proxy a `localhost:1880` que es la dirección que hemos configurado para que atienda peticiones al servicio de Node-RED.

Toda seguridad es poca, por eso, tras el estudio de las opciones de configuración, para tener el control sobre la gente que accede a la aplicación web, habilitamos en los ficheros de configuración un login para el que creamos un usuario administrador capaz de desplegar elementos nuevos y un usuario con permisos de solo lectura por si alguien está interesado en ver el flujo sin riesgo a que aplique modificaciones.

Tras validar el acceso con credenciales, creamos el flujo, representado en la Figura 17, que permitiría la comunicación con el bróker de TTN y posteriormente la inserción de los datos en una tabla de InfluxDB.



**Figura 17.** Flujo de node red.

El nodo de TTN-Ubuntu, que se encuentra en la margen izquierda de la Figura 17, es el encargado de realizar la conexión al bróker de TTN a través de un usuario y la contraseña de la aplicación de TTN, los mensajes son convertidos de un string en formato JSON a su representación en un objeto JavaScript, en el nodo llamado “string to object” de la Figura 17. Esto es necesario para después aplicar sobre ese objeto diferentes filtros, que son los nodos naranjas con una f en la Figura 17, que permiten almacenar sobre diferentes tablas de InfluxDB datos personalizados mediante los nodos de la margen derecha de la Figura 17.

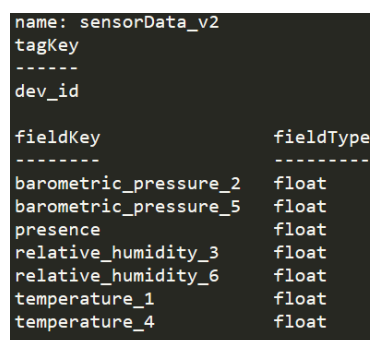
Una vez tenemos los datos que queremos almacenar, es necesario levantar el servicio que nos permitirá hacerlo.

Instalamos InfluxDB, aplicación que también está dotada de un panel de control Web. A diferencia de Node-RED, el panel web de InfluxDB no es necesario para el almacenamiento de datos ni para su consulta. Toda tarea de administración sobre la BBDD se puede realizar desde la línea de comandos solventando así el agujero de seguridad que supone la habilitación del acceso web.

Configuramos la aplicación para que solo funcione por la dirección de localhost y por el puerto 8086 que es su puerto por defecto. De esta manera evitamos que terceras personas se puedan conectar al panel de control web y acceder a los datos almacenados tal y como se representa en la Figura 16.

Como se ha indicado en la explicación del flujo de Node-RED se aplicaron diferentes filtros para clasificar la información a almacenar.

En una tabla se almacenan los datos recogidos por los sensores, se añade además en esa tabla el nombre de los dispositivos como tag para poder realizar consultas específicas sobre cada uno. Esta estructura queda descrita en la Figura 18.



```
name: sensorData_v2
tagKey
-----
dev_id

fieldKey                fieldType
-----
barometric_pressure_2   float
barometric_pressure_5   float
presence                 float
relative_humidity_3      float
relative_humidity_6      float
temperature_1            float
temperature_4            float
```

**Figura 18.** Estructura tabla de datos

En otra tabla se almacenan las coordenadas de los dispositivos para realizar consultas sobre su localización.

Por último, se añadió una tabla adicional en la que no se guarda ningún dato recogido por el dispositivo, esta tabla almacena un valor que representa si el dispositivo está vivo o no. Cada vez que se recibe un mensaje de un dispositivo se almacena una entrada en esta tabla que contiene el Device\_ID del dispositivo y el valor 1. Con una simple consulta podemos saber el estado de los dispositivos y generar un panel de alarmas en Grafana para reportar el mal funcionamiento de alguno de ellos.

Una vez los datos llegan a la BBDD es momento de mostrarlos a través de Grafana.

Tras la instalación y configuración del servidor de Grafana como servicio escuchando en el puerto 3000 al igual que se hizo con los otros dos servicios y que queda representado en la Figura 16.

Como se hizo con Node-Red, es necesario configura Nginx para permitir la entrada de peticiones que vayan dirigidas a grafana. Para ello en caso de que llegue una petición

con URI /grafana, es decir <http://ttnzgapi.ibercivis.es/grafana> hacemos un reverse proxy a localhost:3000 que es la dirección donde escucha este servicio.

Accedimos a través del panel de control web y se configuro un login al igual que se hizo en Node-RED, generando también los dos tipos de usuarios.

Una vez validado el acceso con credenciales creamos las consultas que generan las gráficas, decidimos generar una gráfica por cada tipo de dato reportado por el sensor para facilitar la visualización.

Así pues generaron graficas de presión, humedad, que se puede observar en la Figura 20, y presencia. Para la temperatura se usó un widget de manera que la representación fuera más atractiva (Figura 19).

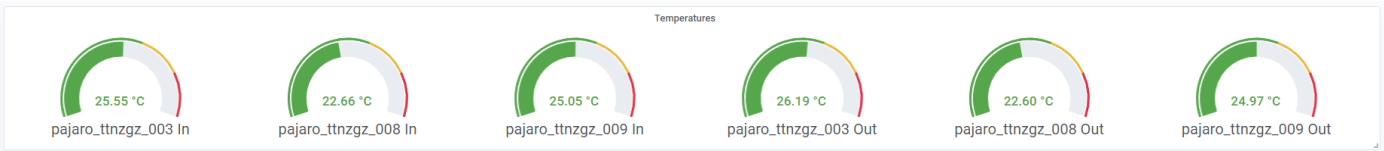


Figura 19. Widget con temperaturas de los dispositivos

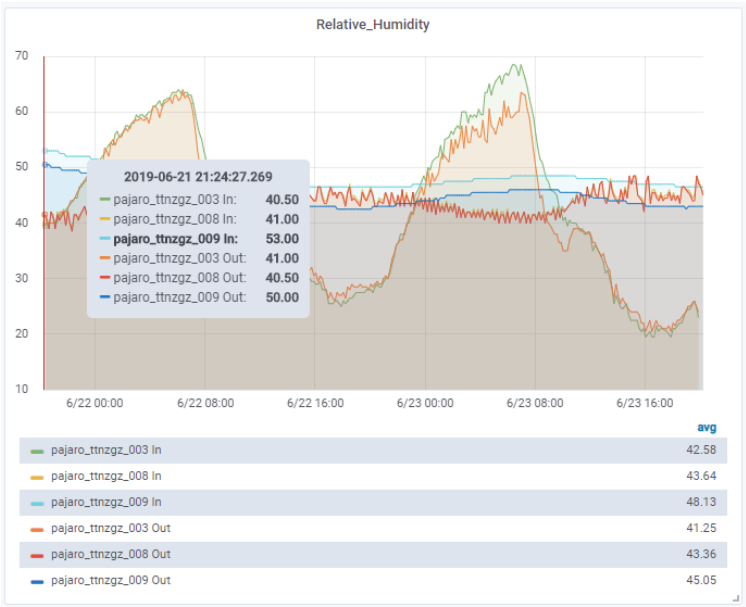
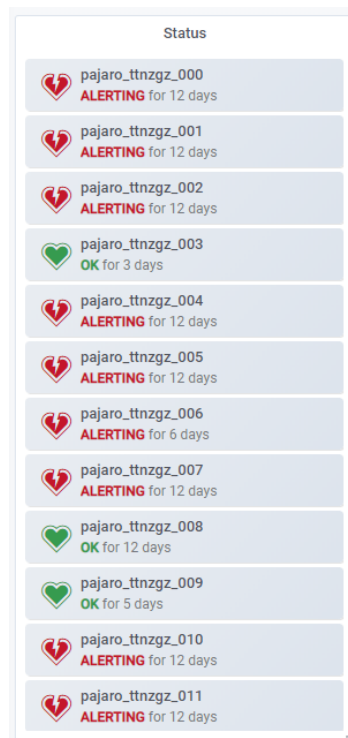


Figura 20. Grafica Humedad Relativa.

Por último generamos un panel de alarmas (Figura 21) que recoge la información del estado de los dispositivos. Este panel reporta un mal funcionamiento en caso de que algún dispositivo no haya dado señales de vida durante la última hora. Este rango de tiempo se estableció debido a que los dispositivos envían mensajes cada 10 minutos, la pérdida de 3 mensajes no supone mucha información, pero la pérdida de 6 mensajes puede ser indicativo de un malfuncionamiento que se debe revisar. En cuanto se recibe una señal de estado vivo se reinicia la cuenta.



**Figura 21.** Panel de alarmas de dispositivos.

Una vez comprobado que todo está funcionando correctamente, se añade una última política a la configuración de Nginx.

En caso de que la petición sea <http://ttnzgapi.ibercivis.es> se redirigirá a la página por defecto de Nginx, para evitar cualquier tipo de información que pueda dar, se ha añadido autenticación a nivel de Nginx.

Con esta solución conseguimos que solo sea necesario tener abiertos a internet el puerto 80 por donde entrará el tráfico web que será después redirigido por el proxy a cada servicio y el puerto 22, necesario para la conexión por SSH.

Como se comentó antes, una vez que todo funciona por el puerto 80 mediante peticiones HTTP, podemos proceder a instalar el certificado que permitirá cifrar las comunicaciones y que dejara el servicio de Nginx que es la puerta de entrada al entorno escuchando por el puerto seguro 443. Para ello hay que indicarle una regla en la configuración de Nginx de que todo el tráfico que venga por el puerto 80 o HTTP sea redirigido al puerto HTTPS o 443 completando el esquema de comunicaciones representado en la Figura 16.

Con esta última modificación obtenemos una comunicación segura hasta el interior de la máquina, evitando que esos ataques automáticos que eran comentados antes puedan encontrar vulnerabilidades en los puertos de la maquina y evitando también que se sepan que servicios están corriendo en nuestro entorno, información que podría usarse para buscar otro tipo de vulnerabilidades.

Para finalizar la implementación de la solución se genera un script que reinicia los servicios de la stack en el orden adecuado para que no surjan dependencias en caso de que sea necesario ejecutarlo por alguna caída de servicio o algún estado inconsistente de algún servicio.



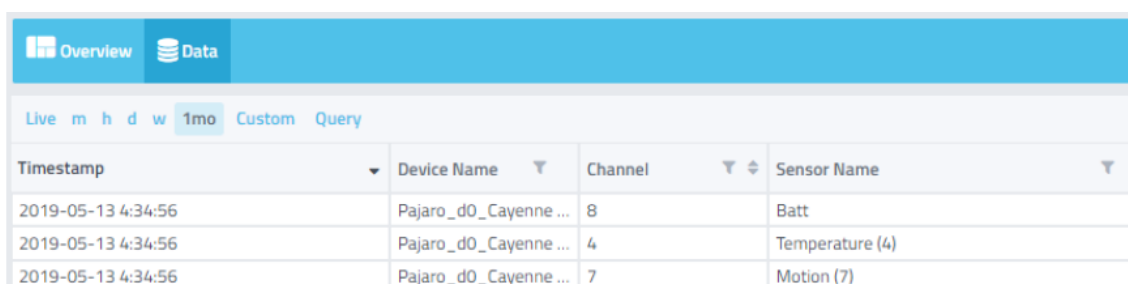
## 6. Validación

Realizada la implementación de los diseños finales, es momento de validar sobre los requisitos originales si se han cumplido o no y como.

Al dividirse los requisitos en dos grupos, se va a comenzar por los requisitos escolares y luego se validarán los de la gestión del proyecto.

### Requisitos escolares

ID	Validado	Demostración
RF1	SI	Cayenne es una herramienta que permite la visualización de los datos de los dispositivos.
RF2	SI	Los dashboards de Cayenne se generan a través de la interfaz web. Apartado 1.C del manual de uso adjuntado en el anexo
RF3	SI	Cayenne es una aplicación accesible a través de internet.
RF4	SI	Cayenne permite crear triggers aplicando acciones en caso de cumplirse condiciones sobre los datos recibidos de los dispositivos.
RF5	SI	A partir de la vista general creada automáticamente con el primer mensaje, el usuario es capaz de modificar el panel a su gusto.
RF6	SI	Existe un apartado llamado Data donde se pueden ver los datos en crudo tal y cual llegan a los canales. Figura 22.

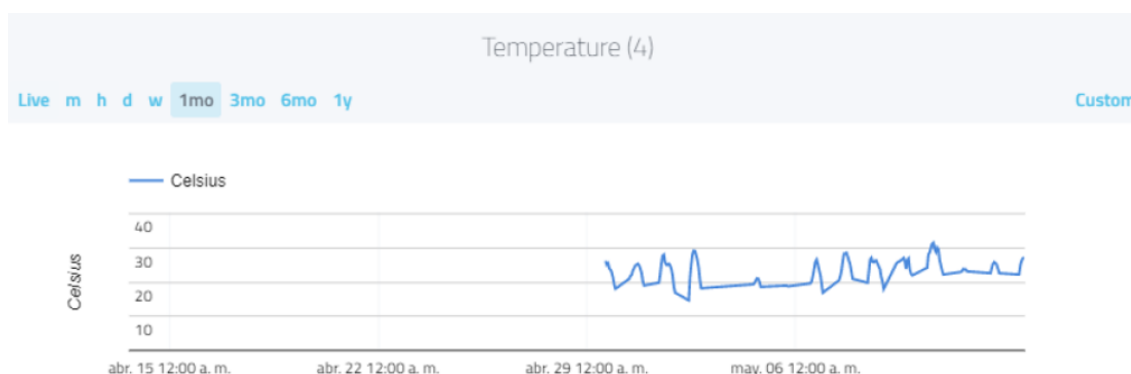


The screenshot shows the Cayenne web interface's 'Data' tab. It features a table with four columns: 'Timestamp', 'Device Name', 'Channel', and 'Sensor Name'. The table contains three rows of data, all from the device 'Pajaro\_d0\_Cayenne ...' at the timestamp '2019-05-13 4:34:56'. The channels are 8 (Batt), 4 (Temperature (4)), and 7 (Motion (7)).

Timestamp	Device Name	Channel	Sensor Name
2019-05-13 4:34:56	Pajaro_d0_Cayenne ...	8	Batt
2019-05-13 4:34:56	Pajaro_d0_Cayenne ...	4	Temperature (4)
2019-05-13 4:34:56	Pajaro_d0_Cayenne ...	7	Motion (7)

**Figura 22.** Datos en crudo de Cayenne

ID	Validado	Demostración
RF7	SI	Cayenne permite generar gráficas y seleccionar el rango de tiempo que se mostrara, Figura 23



**Figura 23.** Grafica Temperatura

ID	Validado	Demostración
RF8	SI	Los datos pueden exportarse en diferentes formatos, CSV, TXT
RNF1	SI	Cayenne tiene disponibles tanto la aplicación Web, como aplicaciones en la Play Store y App Store.
RNF2	SI	Cayenne permite mandar notificaciones tanto a móvil como a direcciones de correo. Además Cayenne puede realizar acciones sobre una URL de GET, POST, PUT...
RNF3	SI	La herramienta utiliza iconos grandes y una interfaz muy limpia, dejando claro las acciones que se pueden realizar, además utiliza muchos iconos para representar unidades que facilitan la vista general.

### Requisitos de gestión del proyecto

ID	Validado	Demostración
RF1	SI	Gracias a Node-RED todos los mensajes enviados pasan a través de los nodos desplegados lo que permite tener control absoluto del flujo de los datos.
RF2	SI	InfluxDB almacena los datos y aplica una política de retención que por defecto es infinita. La BBDD está desplegada sobre un disco de 46 GB, lo que siguiendo el cálculo realizado en la explicación del diseño supone una vida de 60 años al disco. Si por alguna razón externa fuera necesario reducir la información almacenada se podría establecer una política de retención de un tiempo estipulado.
RF3	SI	Además de tener los datos duplicados en Cayenne e InfluxDB, los administradores de Ibercivis se encargan de realizar backups temporales.
RF4	SI	El panel de alarmas permite saber que dispositivos están reportando datos y cuáles no.
RF5	SI	Grafana permite la visualización de los datos mediante gráficas
RF6	SI	Grafana es accesible desde cualquier punto a través de la URL <a href="https://ttnzgapi.ibercivis.es/grafana">https://ttnzgapi.ibercivis.es/grafana</a>
RF7	SI	Grafana permite granular los accesos en Editor, Observador y administrador
RF8	SI	El panel de TTN permite tomar acciones sobre los dispositivos
RNF1	SI	La representación de que el dispositivo está vivo es un corazón verde y de que no responde es un corazón roto y rojo, además los dispositivos están ordenados por id y entran todos dentro de la primera vista sin necesidad de hacer scroll. Gracias a esto de un simple vistazo somos capaces de saber que dispositivos y durante cuánto llevan sin responder. Para obtener más información solo debemos clicar sobre el dispositivo deseado y nos llevara a la gráfica de estado que es más detallada.
RNF2	SI	Aunque Grafana no tenga una aplicación dentro del store, la página web es responsive y se adapta perfectamente a la vista en un dispositivo móvil

Como se puede observar cada diseño cumple los requisitos para el grupo indicado.

Se han validado los requisitos que se plantearon en la fase de análisis pero durante el desarrollo del proyecto han aparecido dos requisitos y que han condicionado la evolución de los diseños.

Uno de ellos eran los costes del diseño que en ambos casos ha quedado en 0 ya que Cayenne es una herramienta gratuita y la máquina virtual es proporcionada por Ibercivis.

El otro, la necesidad de securizar el entorno por el hecho de tener IP publica y necesitar de un acceso web para la visualización de los datos que también se ha cumplido con la instalación del reverse proxy y del certificado HTTPS.



## 7. Conclusiones

Se ha realizado un estudio de diversos diseños, para analizar la viabilidad de implementar alguno de ellos como solución para el problema planteado en los objetivos de la introducción. Esta solución debe permitir la gestión de los dispositivos, del almacenamiento, del flujo de mensajes y de monitorización para que tanto los usuarios como los gestores puedan explotar la información.

Existen 3 tipos de diseños en este análisis, uno basado en el servicio de IoT de AWS apoyado en otros servicios de AWS, el segundo, un diseño basado en servicios de AWS estándar y un diseño basado en una instalación más tradicional dentro de un host de Ibercivis.

Tras el análisis, evolución e implementación de los diseños que cumplen los objetivos del proyecto se puede concluir que, debido al estado en el que se encuentra el proyecto, con solo 15 dispositivos y sin presupuesto, toda implementación basada en servicios cloud solo supone costes. Apoyar un proyecto tan pequeño sobre un servicio tan grande como es la Cloud además de generar costes no produce ninguna ventaja que no pueda suplirse en un modelo más local.

Los alumnos han podido comenzar a trabajar sobre los diferentes proyectos que les llevarán a los paneles de control de Cayenne generados para ellos.

La gestión de los dispositivos se lleva desde la consola de TTN al no ser necesario por el número de dispositivos que tiene el proyecto. Al ser los dispositivos estables en cuanto a que no se están dando de alta/baja de continuo, permite que no sea necesario la automatización de la gestión de estos.

Del almacenamiento y la gestión de los datos se encarga finalmente la Stack de servicios que queda implementada sobre una máquina virtual on-premise.

La visualización de los datos de todos los dispositivos, así como el panel de alarmas que permite saber el estado de las casetas queda implementado dentro de la Stack que incluye la herramienta de visualización Grafana que, junto a un dominio público proporcionado por Ibercivis permite la monitorización de los dispositivos desde cualquier parte.

Con este proyecto se ha podido profundizar en conocimientos sobre el Cloud de Amazon y la utilización de servicios orientados al IoT. También ha servido para reforzar conocimientos adquiridos en asignaturas como Garantía y Seguridad, Seguridad Informática e Ingeniería del Software.

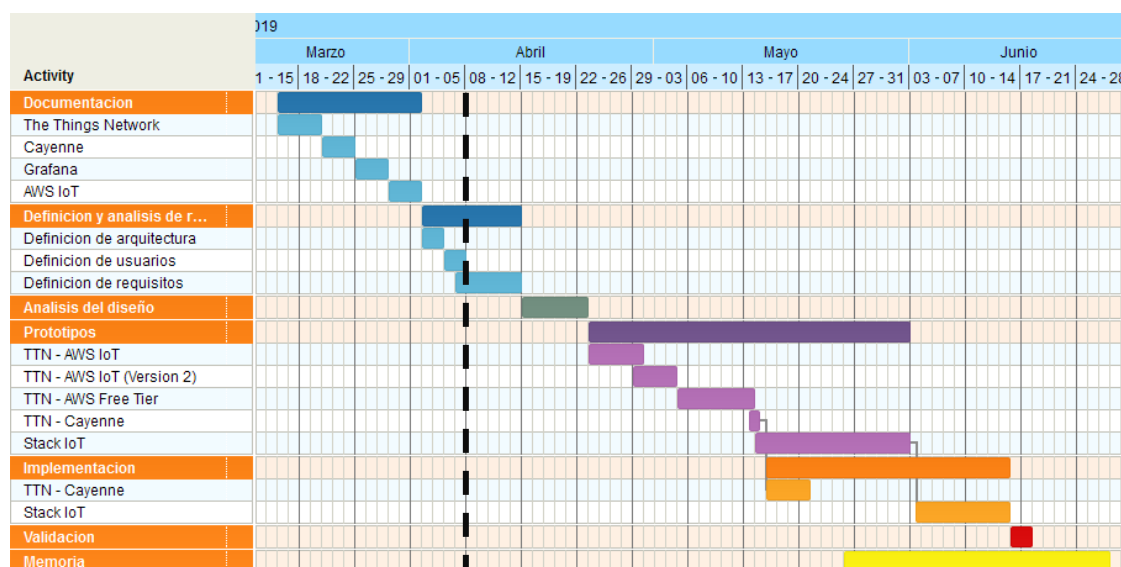
Una continuación de este proyecto podría ser, el estudio de los límites del proyecto a partir de los cuales tendría sentido una implementación de la arquitectura propuesta en la Cloud, permitiendo el auto escalado de instancias en función del crecimiento del

número de dispositivos, así como la gestión del presupuesto proponiendo diseños que redujeran el coste permitiendo automatizar procedimientos repetitivos.

Otra continuación podría ser un análisis más ambicioso que comparara diferentes proveedores cloud para analizar, en caso de existir un presupuesto, cuál de ellos es más adecuado para alojar una solución general de IoT.

A continuación para finalizar la memoria se exponen las horas de esfuerzo y el porcentaje destinado a cada tarea que se ha realizado durante el periodo de realización del proyecto representado en la Figura 24.

Cabe destacar como se ha indicado antes que se realizaron partes del proyecto para la asignatura de Laboratorio de Sistemas Empotrados (LABSE) que se reflejan con la línea a rayas vertical representada en la Figura 24.



**Figura 24.** Diagrama de Gantt con las fechas del proyecto.

El esfuerzo realizado es de 4 horas de media al día, esta media no es del todo real ya que al estar trabajando entre la semana se realizaban una media de 3 horas pero el fin de semana la media superaba las 5 horas.

En este proyecto pueden diferenciarse dos tareas, una labor de ingeniería encargada de la documentación, el estudio, análisis y extracción de conclusiones que forma el 75 % del proyecto y una tarea de implementación más manual y técnica que supone el 25 % de las horas. Este porcentaje se debe a que existe mucho trabajo de puesta en marcha de servicios y diseños que no se considera implementación porque forman parte de las fases de análisis y pruebas y no son parte de la solución final.

El reparto de horas dividido en fases es:

- Documentación: 115 horas en LABSE
- Definición y análisis de requisitos: 60 horas (20 en LABSE + 40 TFG)
- Estudio y análisis de los diseños: 280 horas

- Implementación de la infraestructura: 40 horas
- Puesta en marcha y evaluación de la plataforma: 10 horas
- Elaboración de la documentación: 40 horas
- Publicación de los resultados a los colegios: 6 horas

El total de horas empleadas en la parte del TFG es de 436.

El echo de contar con un año de experiencia laboral como administrador de sistemas Cloud en Everis ha permitido que le numero de horas no se incremente mucho.





## 8. Bibliografía

- [0] Internet de las cosas, Internet of Things o IoT  
[https://es.wikipedia.org/wiki/Internet\\_de\\_las\\_cosas](https://es.wikipedia.org/wiki/Internet_de_las_cosas)
- [1] The Things Network <https://www.thethingsnetwork.org/>
- [2] About LoraWAN, Lora Alliance <https://loro-alliance.org/about-lorawan>
- [3] LoRa <https://en.wikipedia.org/wiki/LoRa>
- [4] Advanced Encryption Standard  
[https://es.wikipedia.org/wiki/Advanced\\_Encryption\\_Standard](https://es.wikipedia.org/wiki/Advanced_Encryption_Standard)
- [5] Cayenne <https://mydevices.com/cayenne/docs/intro/>
- [6] Grafana, <https://grafana.com/>
- [7] AWS CloudWatch, <https://aws.amazon.com/es/cloudwatch/>
- [8] Cloud, <https://www.redhat.com/es/topics/cloud>
- [9] Amazon Web Services, <https://aws.amazon.com/es/>
- [10] AWS IoT, <https://aws.amazon.com/es/iot/>
- [11] Amazon DynamoDB, <https://aws.amazon.com/es/dynamodb/>
- [12] Amazon S3, <https://aws.amazon.com/es/s3/>
- [13] Amazon Kinesis Firehose, <https://aws.amazon.com/es/kinesis/data-firehose/>
- [14] Amazon Athena, <https://aws.amazon.com/es/athena/>
- [15] AWS cli, <https://aws.amazon.com/es/cli/>
- [16] Eclipse Mosquitto, <https://mosquitto.org/>
- [17] Nodejs, <https://nodejs.org/es/>
- [18] Amazon EC2, <https://aws.amazon.com/es/ec2/>
- [19] Security Group, [https://docs.aws.amazon.com/es\\_es/AWSEC2/latest/UserGuide/using-network-security.html](https://docs.aws.amazon.com/es_es/AWSEC2/latest/UserGuide/using-network-security.html)
- [20] AMI, [https://docs.aws.amazon.com/es\\_es/AWSEC2/latest/UserGuide/AMIs.html](https://docs.aws.amazon.com/es_es/AWSEC2/latest/UserGuide/AMIs.html)
- [21] CayenneLPP, <https://mydevices.com/cayenne/docs/lora/#lora-cayenne-low-power-payload>
- [22] NodeRed, <https://nodered.org/>
- [23] InfluxDB, <https://www.influxdata.com/products/influxdb-overview/>
- [24] Nginx, <https://www.nginx.com/>
- [25] SystemD, <https://es.wikipedia.org/wiki/Systemd>



# ANEXO

# Proyecto “Pájaros en la nube”

Manual de uso - herramienta dashboards



## Cayenne

# Índice

## 1. Web

- a. Acceso a la cuenta
- b. Añadir primer dispositivo
- c. Tipo de vistas
  - i. Overview
  - ii. Data

## 2. Móvil

# Web

En este documento se expondrán las posibles acciones a realizar dentro de la página de Cayenne.

Como paso inicial se proporcionará una dirección de correo y una contraseña que servirán para acceder a la cuenta que ya ha sido creada para agilizar los trámites con las credenciales.

## **Acceso a la cuenta**

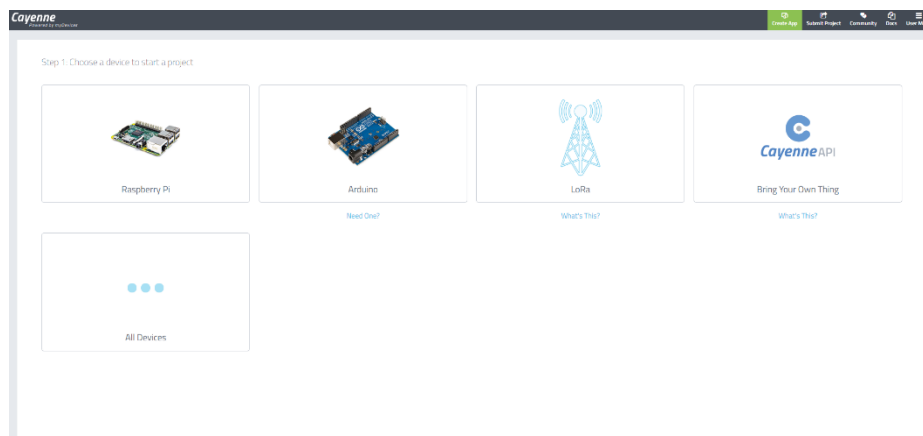
Se utilizará la siguiente URL para acceder al formulario de login:

<https://cayenne.mydevices.com>

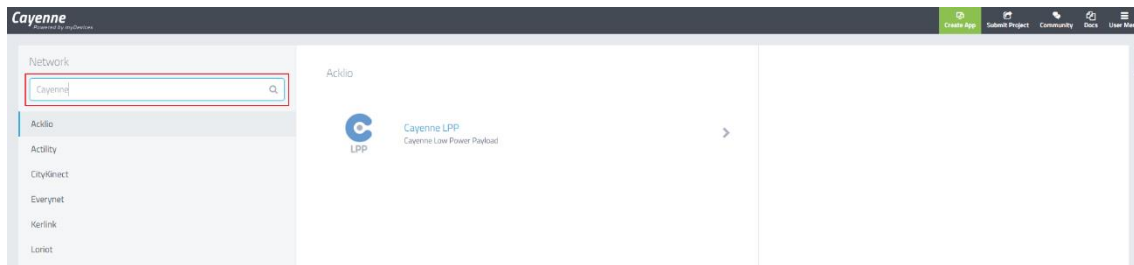
Se ha de completar el usuario y la contraseña con los datos suministrados.

## **Añadir el primer dispositivo**

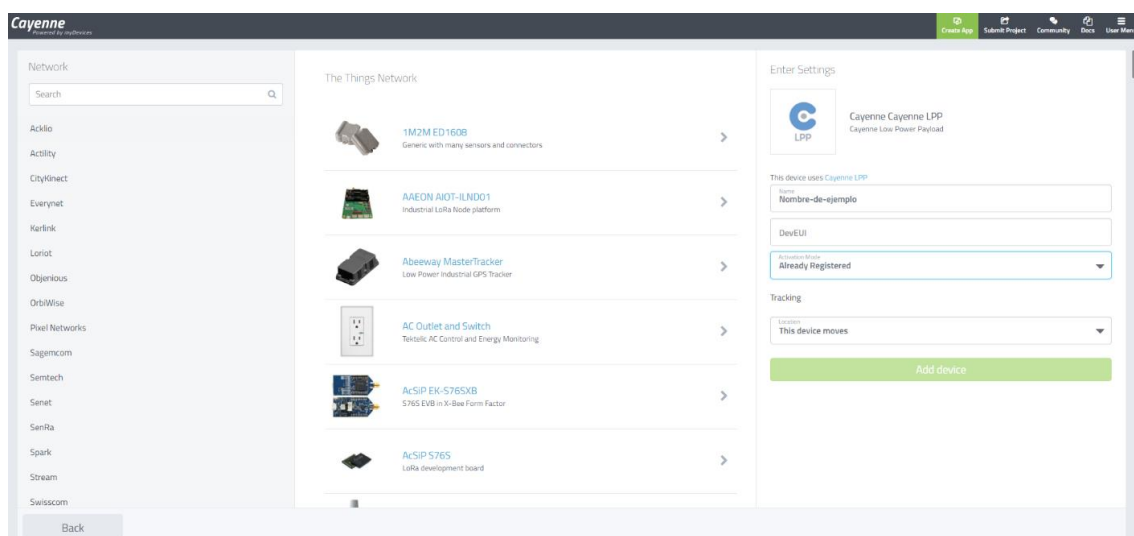
Una vez superado el login se mostrará una página como esta:



Para configurar el dispositivo se seleccionará la opción de “LoRa”, tras esto aparecerán un listado de dispositivos donde se tendrá que buscar la opción “Cayenne LPP” añadiendo en la barra de búsqueda la palabra “Cayenne”.

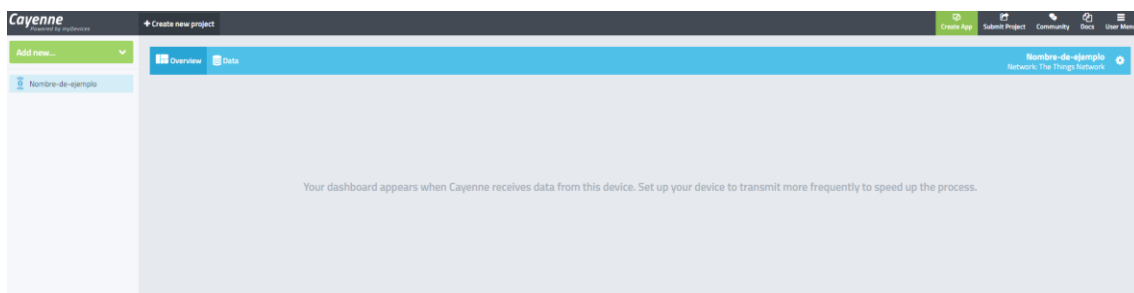


Una vez seleccionada, se desplegará un menú donde se permitirá configurar el nombre del dispositivo y el DevEUI, que es el nombre único del dispositivo que permite a la pagina saber que los datos pertenecen a este dispositivo y no a otro. Este identificador se proporcionará junto a las credenciales para acceder a la cuenta.



Tras configurar esto correctamente, deberían aparecer los dashboards automáticamente.

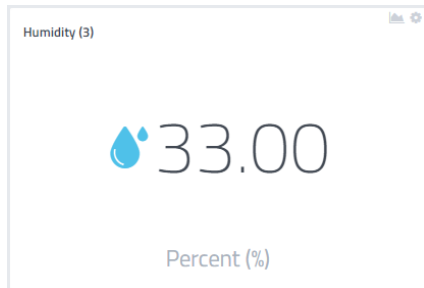
Es posible que la web indique que no se han recibido datos, esto es debido a que se envían cada 10 minutos y por lo tanto podría pasar hasta un periodo de 10 minutos en los que se verá algo como esto.



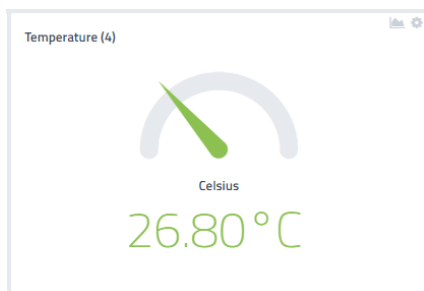
En el momento en que lleguen los primeros datos se generaran los widgets y podremos empezar a interactuar con ellos.

Podemos diferenciar entre 3 tipos de widgets para datos decimales:

- Value: Muestra el valor actual.



- Gauge: Muestra el valor actual sobre el máximo, medio y mínimo (rojo, naranja y verde).



- Line Chart: Muestra el conjunto de valores dentro del periodo de tiempo seleccionado.



Sin embargo para datos digitales (0/1) existen múltiples widgets en los que cada uno de los posibles valores representan un estado. Algunos de estos son:

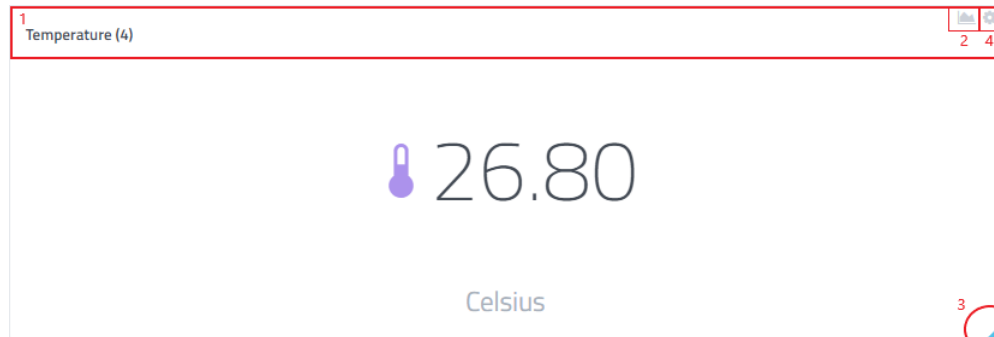
- Detector de movimiento: 0 -> NO, 1 -> SI
- Detector Cerrado/abierto.
- Detector de proximidad: 0 -> NO, 1 -> SI



## Tipos de vistas

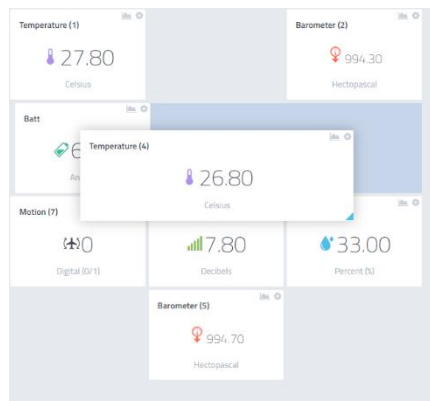
- Overview: Vista general donde se encuentran los widgets con los que se podrá interactuar.

El elemento principal de esta vista es el widget, como se ha mencionado antes hay varios tipos que comparten acciones.



El widget esta formado de 4 elementos interactivos a parte de los datos informativos que contiene.

1. Panel de desplazamiento: Si se coloca el ratón sobre la zona numero 1 se puede mover el widget por toda la vista general



2. Mostrar graficas: Este botón permite ver y descargar desde el histórico anual de datos hasta los mismo que llegan en vivo a la aplicación, todos ellos representados en graficas.



3. Cambiar tamaño: Sirve para redimensionar el widget.
4. Configuración del widget: Abre un menú con diferentes opciones sobre el widget:
  1. Settings: opciones para personalizar el widget:
    - i. Nombre
    - ii. Canal: NO MODIFICAR
    - iii. Widget: Tipo de widget que se han explicado anteriormente
    - iv. Icono: Icono que representa el tipo de datos que se muestran en ese widget.
    - v. Unidad: unidad de medida en que se miden los datos representados
    - vi. Numero de decimales.

2. Trigger: Permite programar eventos cuando el valor seleccionado cumpla una condición, estos eventos son notificaciones a uno o varios números de teléfonos, direcciones de correo o consultas sobre una página web.

Se ha de arrastrar el dispositivo y elegir una de las métricas, establecer la condición que lanzara la acción y configurar esta acción.

- Data: Se pueden ver los datos en crudo del periodo que se seleccione.

Pajaro_d0_Cayenne LPP Network: The Things Network									
Live m h d w 1mo Custom Query									
Timestamp	Device Name	Channel	Sensor Name	Sensor ID	Data Type	Unit	Values		
2019-05-13 4:34:56	Pajaro_d0_Cayenne ...	8	Batt	b42b56f0-6a83-11e9-b189-ab9cb6660d7e	analog_sensor	null	6.0799999237061		
2019-05-13 4:34:56	Pajaro_d0_Cayenne ...	4	Temperature (4)	b3d71c70-6a83-11e9-bdb5-dfd20f02ea3f	temp	c	26.7999999237061		
2019-05-13 4:34:56	Pajaro_d0_Cayenne ...	7	Motion (7)	b4271130-6a83-11e9-bdb5-dfd20f02ea3f	motion	d			
2019-05-13 4:34:56	Pajaro_d0_Cayenne ...	6	Humidity (6)	b47b5160-6a83-11e9-bdb5-dfd20f02ea3f	rel_hum	p	25		
2019-05-13 4:34:56	Pajaro_d0_Cayenne ...	1	Temperature (1)	b3aab540-6a83-11e9-9dd7-93e53f1934d3	temp	c	27.7999999237061		
2019-05-13 4:34:56	Pajaro_d0_Cayenne ...	3	Humidity (3)	b3be6450-6a83-11e9-bdb5-dfd20f02ea3f	rel_hum	p	33		
2019-05-13 4:34:56	Pajaro_d0_Cayenne ...	2	Barometer (2)	b3b9f780-6a83-11e9-9dd7-93e53f1934d3	bp	hpa	994.29998779297		
2019-05-13 4:34:56	Pajaro_d0_Cayenne ...	101	SNR	b33229e0-6a83-11e9-933e-cf08617625ed	snr	db	7.8000001907349		
2019-05-13 4:34:56	Pajaro_d0_Cayenne ...	100	RSSI	b31a0e00-6a83-11e9-933e-cf08617625ed	rssi	dbm	-104		
2019-05-13 4:34:56	Pajaro_d0_Cayenne ...	5	Barometer (5)	b3debd90-6a83-11e9-b189-ab9cb6660d7e	bp	hpa	994.70001220703		

# Móvil

Cayenne tiene la posibilidad de utilizarse en el móvil a través de la aplicación que está disponible tanto para Android como para IOS.

La aplicación se llama Cayenne de la empresa MyDevices y tiene funcionalidades similares a la aplicación web.

Para acceder solo se ha de completar el usuario y contraseña igual que en la web seleccionando el botón que dice “Sign into my account”.



Una vez pasada la autenticación aparecerán los widgets igual que se encuentran en la aplicación web.

